WEB PAGES 10,418,340 pages found. Family Filter is off

1. **Sandia National Laboratories - HTML Reference Manual (removed)**
   HTML Reference Manual (removed) Neither the main page nor any of the companion sub-pages of Sandia National Laboratories' HTML Reference Manual had...
   URL: www.sandia.gov/sci_compute/html_ref.html
   Last modified on: 11-Jan-1999 - 5K bytes - in English
   [Translate] [More pages from this site] [Related pages]

2. **PHP: PHP Manual Quick Reference**
   PHP Manual Quick Reference. Here is a list of all the PHP 3.0 functions. Click on any one of them to jump to that page in the manual. abs acos...
   URL: www.php.net/quickref.php3
   Last modified on: 21-Oct-1999 - 76K bytes - in English
   [Translate] [More pages from this site] [Related pages] [Facts about: **Circle Net Inc**]

3. **No Title**
   MIRROR(1L) MISC. REFERENCE MANUAL PAGES MIRROR(1L) NAME mirror - mirror packages on remote sites SYNOPSIS mirror -[dvTn] [-Ufilename...
   URL: nic.funet.fi/FUNET/hamster/mirror.txt
   Last modified on: 2-Dec-1993 - 19K bytes - in English
   [Translate] [Related pages]

4. **No Title**
   MIRROR(1L) MISC. REFERENCE MANUAL PAGES MIRROR(1L) NAME mirror - mirror packages on remote sites SYNOPSIS mirror -[dvTn] [-Ufilename...
   URL: ftp.funet.fi/funet/hamster/mirror.txt
   Last modified on: 2-Dec-1993 - 19K bytes - in English
   [Translate] [Related pages]

5. **The Registrar Reference Manual**
   Browse. Directories. OUR Newswire. Administrative Forms. Course Summary. ISIS Screens & Procedures. OUR Departments. Registrar Documentation....
   URL: www.our.psu.edu/
   Last modified on: 25-Jan-2000 - 7K bytes - in English
   [Translate] [More pages from this site] [Related pages] [Facts about: **Pennsylvania Stat...**]

6. **Space Shuttle News Reference Manual**

alta vista: SEARCH | Search | Live! | Shopping | Raging Bull | Free Internet Access | Email

Web Results

Find this: ftpget+article 3893 of comp.lang.perl | Search

Tip: Use quotes for multi-word phrases.

Language: any language

Help | Family Filter | Language Settings | Advanced Web Search

Click a tab for more results on ftpget+article 3893 of comp.... What is a tab?

Products | News | Discussion Groups | The Web | Images | MP3/Audio | Video | Directory

WEB PAGES 35,504 pages found. Family Filter is off

1. No Title
   Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
   URL: www.metronet.com/perlinfo/scripts/ftpstuff/ftpget
   Last modified on: 1-Jul-1993 - 22K bytes - in English
   [Translate] [More pages from this site] [Related pages]

2. No Title
   Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
   URL: ftp.loxinfo.co.th/pub/unix/programming/perl/ftp/ftpget
   Last modified on: 1-Jul-1993 - 22K bytes - in English
   [Translate] [Related pages]

3. No Title
   Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
   URL: kawnug.oznet.ksu.edu/perl/SCRIPTS/ftpstuff/ftpget
   Last modified on: 27-Mar-1996 - 22K bytes - in English
   [Translate] [More pages from this site] [Related pages] [Facts about: Kansas State Univ...]

4. Comprehensive Perl Archive Network
   CPAN: Comprehensive Perl Archive Network. Welcome to CPAN! Here you will find All Things Perl. CPAN is the Comprehensive Perl Archive Network....
   URL: www.cpan.org/
   Last modified on: 1-Nov-1999 - 3K bytes - in English
   [Translate] [More pages from this site] [Related pages]

5. comp.lang.perl.tk FAQ
   From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
   URL: w4.lns.cornell.edu/~pvhp/ptk/ptkFAQ.html
   Last modified on: 27-Jul-1997 - 251K bytes - in English
   [Translate] [More pages from this site] [Related pages]

6. comp.lang.perl.tk FAQ
   From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
   URL: www.lns.cornell.edu/~pvhp/ptk/ptkTOC.html
   Last modified on: 27-Jul-1997 - 12K bytes - in English
   [Translate] [More pages from this site] [Related pages]

Web Results

Boolean query: mirror and perl

Search | Help | Family Filter | Language Settings

Tip: Include applet:[text] in your search. | About Advanced Se

Sort by:

Language: any language | Show one result per Web site

From: 31/1/90 To: 31/5/94 (e.g. 31/12/99) | Main Se

WEB PAGES 105 pages found. | Family Filter is off

11. No Title
Revision 1.30 1993/06/18 16:55:51 lmjm # Drop the uses of $con. # # Revision 1.29 1993/06/04 15:20:34 lmjm # Patch by Joergen Haegg. to spot when no...
URL: irs.cac.psu.edu/IRS/cgi/changelogs-rev1/ftp
Last modified on: 28-Jun-1993 - 5K bytes - in English
[Translate][More pages from this site][**Company factsheet**]

12. Where can I get more information on Perl?
Where can I get more information on Perl? We'll cover five areas here: USENET (where you're probably reading this), publications, the reference...
URL: www.cis.ufl.edu/perl/faq/html/1.7.html
Last modified on: 11-Feb-1994 - 6K bytes - in English
[Translate][More pages from this site][**Company factsheet**]

13. No Title
text -*- Last modified Tue Mar 30 18:10:57 1993 This is an incomplete and probably out-of-date list of all the packages distributed on...
URL: ion.apana.org.au/pub/gnu/DESCRIPTIONS
Last modified on: 3-Jan-1994 - 36K bytes - in English
[Translate]

14. Index of /Icons/Alcons/READMEs
Index of /Icons/Alcons/READMEs. <PLAINTEXT> Anthony's X Icon Library (version 1.5) ---------- ---------- Main ReadMe File...
URL: shiva.di.uminho.pt/Icons/Alcons/READMEs
Last modified on: 18-Feb-1994 - 7K bytes - in English
[Translate]

15. No Title
usr/bin/perl # Mirror Master. # Run several mirrors in parallel. # # By Lee McLoughlin. # You can do what you like with this except claim that you...
URL: strucbio.biologie.uni-konstanz.de/pdb/mirror/mm
Last modified on: 18-Jan-1994 - 10K bytes - in English
[Translate][More pages from this site]

16. No Title
This is the file README for the gzip distribution, version 1.2.4. gzip (GNU zip) is a compression utility designed to be a replacement for...
URL: www.centralcoastdata.org/WaterData/helpers/gzip/readme.txt

WEB PAGES 7 pages found. Family Filter is off

1. **No Title**
   Article 5397 of comp.lang.perl: Xref: feenix.metronet.com comp.infosystems.www:1336 comp.lang.perl:5397 Newsgroups:...
   URL: ftp.telecom.sk/pub/mirror/CPAN/scripts/i...WWW/http.get.pl
   Last modified on: 25-Aug-1993 - 2K bytes - in English
   [Translate]

2. **No Title**
   Hypertext Transfer Protocol (HTTP) Tim Berners-Lee, CERN Internet Draft Expires 5 May 1994 5 Nov 1993 Hypertext Transfer Protocol (HTTP) A Stateless...
   URL: freenet.msp.mn.us/help.sav/htmldocs/http-spec.txt
   Last modified on: 20-Nov-1993 - 65K bytes - in English
   [Translate] [Related pages]

3. **No Title**
   Hypertext Transfer Protocol (HTTP) Tim Berners-Lee, CERN Internet Draft Expires 5 May 1994 5 Nov 1993 Hypertext Transfer Protocol (HTTP) A Stateless...
   URL: ncstrl.informatik.uni-stuttgart.de/fis/http-spec.txt
   Last modified on: 31-Mar-1994 - 65K bytes - in English
   [Translate] [Related pages]

4. **Lynx Users Guide v2.2**
   Lynx Users Guide Version 2.2. Lynx is a fully-featured World Wide Web (WWW) client for users running cursor-addressable, character-cell display...
   URL: www.cs.tufts.edu/lynx/Lynx_users_guide.html
   Last modified on: 16-Mar-1994 - 31K bytes - in English
   [Translate] [Related pages] [**Company factsheet**]

5. **No Title**
   local/bin/perlbin/perl # # urlget — get a document given a WWW URL # # Jack Lund 9/3/93 zippy@ccwf.cc.utexas.edu # # from hget by: # Oscar...
   URL: www.chemie.uni-dortmund.de/~loki/exp/urlget
   Last modified on: 23-Mar-1994 - 4K bytes - in English
   [Translate] [Related pages]

6. **No Title**
   Hypertext Transfer Protocol (HTTP) Tim Berners-Lee, CERN Internet Draft Expires 5 May 1994 5 Nov 1993 Hypertext Transfer Protocol (HTTP) A Stateless...

WEB PAGES 81 pages found. Family Filter is off

1. **No Title**
   Computer underground Digest Sun Oct 25, 1992 Volume 4 : Issue 53 Editors: Jim Thomas and Gordon Meyer (TK0JUT2@NIU.BITNET) Archivist: Brendan Kehoe...
   URL: venus.soci.niu.edu/~cudigest/CUDS4/cud453.txt
   Last modified on: 7-Nov-1992 - 46K bytes - in English
   [Translate] [Related pages]

2. **No Title**
   Article 5397 of comp.lang.perl: Xref: feenix.metronet.com comp.infosystems.www:1336 comp.lang.perl:5397 Newsgroups:...
   URL: ftp.telecom.sk/pub/mirror/CPAN/scripts/i...WWW/http.get.pl
   Last modified on: 25-Aug-1993 - 2K bytes - in English
   [Translate]

3. **Perl-FAQ-1**
   Table of Contents. 1.1) What is Perl? 1.2) Is Perl hard to learn? 1.3) Should I program everything in Perl? 1.4) Where can I get Perl over the...
   URL: ftp.sra.co.jp/public/doc/perl-faq/Perl-FAQ-1.html
   Last modified on: 27-Jan-1994 - 37K bytes - in English
   [Translate] [Related pages]

4. **No Title**
   usr/bin/perl # Obey the remove commands generated but not done by mirror. # NEED TO unlink /public/micros/ibmpc/simtel20/ prdsht/tc810.arc # NEED TO...
   URL: www.mit.edu/afs/sipb.mit.edu/user/warlor...rror/do_unlinks
   Last modified on: 9-Aug-1993 - 316 bytes - in English
   [Translate] [Related pages] [**Company factsheet**]

5. **No Title**
   Archive-name: finding-sources Version: $Id: csw_faq,v 1.109 1993/03/03 14:32:11 jik Exp $ I. Table of contents This article contains the following...
   URL: ftp.kiarchive.ru/pub/internet/faq/csw_faq
   Last modified on: 28-Mar-1993 - 42K bytes - in English (KOI8-R)
   [Translate] [Related pages]

6. **How can I compare two date strings?**
   How can I compare two date strings? If the dates are in an easily parsed, predetermined format, then you can break

WEB PAGES 35,504 pages found. Family Filter is off

1. **No Title**
Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
URL: www.metronet.com/perlinfo/scripts/ftpstuff/ftpget
Last modified on: 1-Jul-1993 - 22K bytes - in English
[Translate] [More pages from this site] [Related pages]

2. **No Title**
Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
URL: ftp.loxinfo.co.th/pub/unix/programming/perl/ftp/ftpget
Last modified on: 1-Jul-1993 - 22K bytes - in English
[Translate] [Related pages]

3. **No Title**
Article 3893 of comp.lang.perl: Xref: feenix.metronet.com comp.lang.perl:3893 Path:...
URL: kawnug.oznet.ksu.edu/perl/SCRIPTS/ftpstuff/ftpget
Last modified on: 27-Mar-1996 - 22K bytes - in English
[Translate] [More pages from this site] [Related pages] [Facts about: **Kansas State Univ...**]

4. **Comprehensive Perl Archive Network**
CPAN: Comprehensive Perl Archive Network. Welcome to CPAN! Here you will find All Things Perl. CPAN is the Comprehensive Perl Archive Network....
URL: www.cpan.org/
Last modified on: 1-Nov-1999 - 3K bytes - in English
[Translate] [More pages from this site] [Related pages]

5. **comp.lang.perl.tk FAQ**
From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
URL: w4.lns.cornell.edu/~pvhp/ptk/ptkFAQ.html
Last modified on: 27-Jul-1997 - 251K bytes - in English
[Translate] [More pages from this site] [Related pages]

6. **comp.lang.perl.tk FAQ**
From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
URL: www.lns.cornell.edu/~pvhp/ptk/ptkTOC.html
Last modified on: 27-Jul-1997 - 12K bytes - in English
[Translate] [More pages from this site] [Related pages]

Web Results

Boolean query: mirror and perl

Search | Help | Family Filter | Language Settings

Example: (marketing OR sales) AND business

Preview our NEW Advanced Search Ce

Sort by:

Language: any language | Show one result per Web site

From: 31/1/90 To: 31/3/94 (e.g. 31/12/99) Main Se

WEB PAGES 81 pages found. Family Filter is off

1. **No Title**
   Computer underground Digest Sun Oct 25, 1992 Volume 4 : Issue 53 Editors: Jim Thomas and Gordon Meyer (TK0JUT2@NIU.BITNET) Archivist: Brendan Kehoe...
   URL: venus.soci.niu.edu/~cudigest/CUDS4/cud453.txt
   Last modified on: 7-Nov-1992 - 46K bytes - in English
   [Translate] [Related pages]

2. **No Title**
   Article 5397 of comp.lang.perl: Xref: feenix.metronet.com comp.infosystems.www:1336 comp.lang.perl:5397 Newsgroups:...
   URL: ftp.telecom.sk/pub/mirror/CPAN/scripts/i...WWW/http.get.pl
   Last modified on: 25-Aug-1993 - 2K bytes - in English
   [Translate]

3. **Perl-FAQ-1**
   Table of Contents. 1.1) What is Perl? 1.2) Is Perl hard to learn? 1.3) Should I program everything in Perl? 1.4) Where can I get Perl over the...
   URL: ftp.sra.co.jp/public/doc/perl-faq/Perl-FAQ-1.html
   Last modified on: 27-Jan-1994 - 37K bytes - in English
   [Translate] [Related pages]

4. **No Title**
   usr/bin/perl # Obey the remove commands generated but not done by mirror. # NEED TO unlink /public/micros/ibmpc/simtel20/ prdsht/tc810.arc # NEED TO...
   URL: www.mit.edu/afs/sipb.mit.edu/user/warlor...rror/do_unlinks
   Last modified on: 9-Aug-1993 - 316 bytes - in English
   [Translate] [Related pages] [**Company factsheet**]

5. **No Title**
   Archive-name: finding-sources Version: $Id: csw_faq,v 1.109 1993/03/03 14:32:11 jik Exp $ I. Table of contents This article contains the following...
   URL: ftp.kiarchive.ru/pub/internet/faq/csw_faq
   Last modified on: 28-Mar-1993 - 42K bytes - in English (KOI8-R)
   [Translate] [Related pages]

6. **How can I compare two date strings?**
   How can I compare two date strings? If the dates are in an easily parsed, predetermined format, then you can break

WEB PAGES 19,812 pages found. Family Filter is off

1. **No Title**
   bin/sh #From cs.utexas.edu!asuvax!gatech!ne s.byu.edu!eff!iWarp.intel.com| ews Fri May 1 14:52:11 CDT 1992 #Article: 10383 of comp.lang.perl #Path:...
   URL: www.metronet.com/perl/scripts/ftpstuff/ftpr
   Last modified on: 6-Apr-1993 - 8K bytes - in English
   [Translate] [More pages from this site] [Related pages]

2. **Comprehensive Perl Archive Network**
   CPAN: Comprehensive Perl Archive Network. Welcome to CPAN! Here you will find All Things Perl. CPAN is the Comprehensive Perl Archive Network....
   URL: www.cpan.org/
   Last modified on: 1-Nov-1999 - 3K bytes - in English
   [Translate] [More pages from this site] [Related pages]

3. **comp.lang.perl.tk FAQ**
   From. pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
   URL: w4.lns.cornell.edu/~pvhp/ptk/ptkFAQ.html
   Last modified on: 27-Jul-1997 - 251K bytes - in English
   [Translate] [More pages from this site] [Related pages]

4. **comp.lang.perl.tk FAQ**
   From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
   URL: www.lns.cornell.edu/~pvhp/ptk/ptkTOC.html
   Last modified on: 27-Jul-1997 - 12K bytes - in English
   [Translate] [More pages from this site] [Related pages]

5. **PHOAKS: Frequency for comp.lang.perl.announce**
   People Helping One Another Know Stuff. Freq. "Together, we know it all." ;POSTER_ID=;" TARGET=_top>Pheedback. Recency. Top Posters. Area Summary....
   URL: www.phoaks.com/phoaks2/newsgroups/comp/l...resources0.html
   Last modified on: 3-Nov-1999 - 91K bytes - in English
   [Translate] [More pages from this site] [Related pages]

6. **CGI Scripts and Resources**
   CGI scripts, desktop publishing, fonts, web authoring, and much more...
   URL: desktoppublishing.com/cgi.html
   Last modified on: 7-Jan-2000 - 13K bytes - in English

WEB PAGES » 19,812 pages found. » Family Filter is off

1. **No Title**
bin/sh #From cs.utexas.edu!asuvax!gatech!ne s.byu.edu!eff!iWarp.intel.com| ews Fri May 1 14:52:11 CDT 1992 #Article: 10383 of comp.lang.perl #Path:...
URL: www.metronet.com/perl/scripts/ftpstuff/ftpr
Last modified on: 6-Apr-1993 - 8K bytes - in English
[Translate] [More pages from this site] [Related pages]

2. **Comprehensive Perl Archive Network**
CPAN: Comprehensive Perl Archive Network. Welcome to CPAN! Here you will find All Things Perl. CPAN is the Comprehensive Perl Archive Network....
URL: www.cpan.org/
Last modified on: 1-Nov-1999 - 3K bytes - in English
[Translate] [More pages from this site] [Related pages]

3. **comp.lang.perl.tk FAQ**
From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
URL: w4.lns.cornell.edu/~pvhp/ptk/ptkFAQ.html
Last modified on: 27-Jul-1997 - 251K bytes - in English
[Translate] [More pages from this site] [Related pages]

4. **comp.lang.perl.tk FAQ**
From: pvhp@lns62.lns.cornell.edu (Peter Prymmer) Newsgroups: comp.lang.perl.tk,comp.lang.pe l.announce,comp.answers,news.a swers Subject:...
URL: www.lns.cornell.edu/~pvhp/ptk/ptkTOC.html
Last modified on: 27-Jul-1997 - 12K bytes - in English
[Translate] [More pages from this site] [Related pages]

5. **PHOAKS: Frequency for comp.lang.perl.announce**
People Helping One Another Know Stuff. Freq. "Together, we know it all." ;POSTER_ID=;" TARGET=_top>Pheedback. Recency. Top Posters. Area Summary....
URL: www.phoaks.com/phoaks2/newsgroups/comp/l...resources0.html
Last modified on: 3-Nov-1999 - 91K bytes - in English
[Translate] [More pages from this site] [Related pages]

6. **CGI Scripts and Resources**
CGI scripts, desktop publishing, fonts, web authoring, and much more...
URL: desktoppublishing.com/cgi.html
Last modified on: 7-Jan-2000 - 13K bytes - in English

(FILE 'USPAT' ENTERED AT 17:28:07 ON 14 JUL 1999)

L1	0	S	BROADCAST? AND 4974149/PN
L2	2247	S	BROADCAST? (P) SELECTIVE?
L3	2	S	L2 AND 4974149/UREF
L4	70	S	L2 (P) UPDAT?
L5	4	S	L4 (P) HOST
L6	8	S	L4 (P) REMOTE?
L7	349	S	BROADCAST? (P) (SELECTIVE? (2A) (UPDAT? OR RECEIV?))
L8	26	S	L7 (P) REMOTE?
L9	396	S	BROADCAST? (4A) UPDAT?
L10	23	S	L9 (P) SELECTIVE?

```
     (FILE 'USPAT' ENTERED AT 13:48:26 ON 14 JUL 1999)
L1            1 S 4974149/PN
L2            0 S L1 (P) MODIF?
L3            0 S L1 (P) CHANG?
L4          241 S (ALTER? OR MODIF? OR CHANG?) (2A) DESCRIPTOR?
L5          113 S L4 (P) DATA
L6           12 S L5 (P) REQUEST?
L7            0 S L1 (P) DESCRIPTOR?
L8            1 S L1 AND ((MODIF? OR CHANG? OR CREAT?) (2A) DESCRIPTOR?)
L9            1 S L1 AND CENTRAL DIGITAL
L10          23 S USER (4A) (TRANSMIT? OR SEND? OR SENT OR PROVIDE?) (4A)
(DE
L11          45 S (SENT OR SEND? OR PROVID? OR TRANSFER? OR TRANSMIT?) (4A
) (
L12           9 S L11/AB,CLM
```

(FILE 'USPAT' ENTERED AT 12:48:35 ON 13 JUL 1999)

L1 715 S (REQUEST? (4A) (INFORMATION OR DATA OR UPDATE)) (4A) REMOTE

L2 43 S L1 (P) SOFTWARE

L3 1499 S (REQUEST? (4A) (INFORMATION OR DATA OR UPDATE)) (P) ((REMOT

L4 90 S L3 (P) ADDRESS? (P) SPECIF?

L5 314 S META-DATA OR METADATA

L6 0 S L1 (P) L5

L7 32 S L5 (4A) (TRANSFER? OR TRANSMIT? OR SEND? OR SENT)

L8 2 S L7 (P) REMOTE?

L9 70 S L5 (P) UPDAT?

L10 5 S L7/AB,CLM

```
      (FILE 'USPAT' ENTERED AT 08:19:55 ON 13 JUL 1999)
L1         74088 S (TRANSFER? OR SEND OR TRANSMIT?) (4A) (INSTRUCTIONS OR S
PEC
L2           194 S L1 (P) UPDAT? (P) SOFTWARE
L3             0 S L2 (P) (INTERNET OR WEB)
L4           671 S L1 (P) (INTERNET OR WEB)
L5            21 S L4 (P) UPDAT?
L6             3 S L2 (P) (CD-ROM OR CDROM)
L7           687 S L1 (P) REQUEST? (P) UPDAT?
L8          6486 S L1 (P) REMOTE
L9          1321 S L8 (P) CENTRAL?
L10          327 S (AUTOMATIC? (4A) UPDAT?) (P) (REMOTE? OR CENTRAL?)
L11           20 S L10 (P) L1
L12           11 S 5694546/UREF
```

_□s_□p_□l_□i_□t__□c_□h_□u_□n_□k Size of chunks to split up files into [102400]

_□s_□p_□l_□i_□t__□p_□a_□t_□t regexp of remote pathnames to split up before storing locally

_□l_□o_□c_□a_□l__□l_□s__□l_□R__□f_□i_□l_□e
local file containing ls-lR - else use remote ls_lR_file. This is useful when first mirroring a large package.

_□l_□s__□l_□R__□f_□i_□l_□e remote file containing ls-lR - else run remote ls

_□n_□a_□m_□e__□m_□a_□p_□p_□i_□n_□g_□s remote to local pathname mappings (a perl _□s command, eg s:old:new:) currently only one allowed

_□g_□e_□t__□n_□e_□w_□e_□r get the remote file if its date is newer than local [true]

_□g_□e_□t__□s_□i_□z_□e__□c_□h_□a_□n_□g_□e
get the file if size if different from local. If a file is compressed when fetched then the size is automatically ignored. [true]

_□c_□o_□m_□p_□r_□e_□s_□s__□p_□a_□t_□t regexp of files to compress before storing locally. (See also get_size_change.)

_□c_□o_□m_□p_□r_□e_□s_□s__□e_□x_□c_□l regexp of files not to compress [\.[zZ]]

_□f_□o_□r_□c_□e__□t_□i_□m_□e_□s Force local times to match remote times [yes]

_□r_□e_□t_□r_□y__□c_□a_□l_□l If initial connect fails retry ONCE after ONE minute [yes]

_□u_□p_□d_□a_□t_□e__□l_□o_□g Filename, relative to local_dir, where an update report is to be kept

_□m_□a_□i_□l__□t_□o Mail a report to this comma separated list of people

_□u_□s_□e_□r User name or uid to give to local pathnames

_□g_□r_□o_□u_□p Group name or gid to give to local pathnames

_□f_□i_□l_□e__□m_□o_□d_□e Mode to give files created locally [0444]

_□d_□i_□r__□m_□o_□d_□e mode to give directories created locally [0755]

_□t_□i_□m_□e_□o_□u_□t timeout ftp requests after this many seconds

[20]

_□f□t_□p__□p_□o_□r_□t port number of remote ftp daemon [21]

_□p□r_□o_□x_□y set to 1 to use proxy ftp service [0]

_□p_□r_□o_□x_□y__□f_□t_□p__□p_□o_□r_□t port number of proxy-service ftp daemon [4514]

_□p□r_□o_□x_□y__□g_□a_□t_□e_□w_□a_□y name of proxy-service, may also be supplied by environmental variable INTERNET_HOST [internet-gateway]

_□r_□e_□c_□u_□r_□s_□i_□v_□e do sub directories as well [true]

_□f_□l_□a_□g_□s__□r_□e_□c_□u_□r_□s_□i_□v_□e
flags to send to ls to do a recursive listing

_□f_□l_□a_□g_□s__□n_□o_□n_□r_□e_□c_□u_□r_□s_□i_□v_□e
flags to send to ls to do a non-recursive listing

_□m_□o_□d_□e__□c_□o_□p_□y flag indicating if we need to copy the mode bits [false]

_□i_□n_□t_□e_□r_□a_□c_□t_□i_□v_□e noninteractive copy default [false]

_□t_□e_□x_□t__□m_□o_□d_□e transfer in binary mode by default [false]

_□f_□o_□r_□c_□e transfer selectively by default [false]

_□g_□e_□t_□f_□i_□l_□e perform get, not put by default [true]

_□v_□e_□r_□b_□o_□s_□e Verbose messages [false]

_□d_□i_□s_□c_□o_□n_□n_□e_□c_□t disconnect from remote site at end of package [false]

*_□r_□e_□m_□o_□t_□e__□f_□s Remote file store type. (Only copes with unix at the moment) [unix]

_□m_□a_□i_□l__□p_□r_□o_□g Program called to send to the mail_to list. [mail]

_□d_□e_□l_□e_□t_□e__□s_□o_□u_□r_□c_□e Delete the source files and dirs once transfered. [false]

Each group of keywords defines how to mirror a particular package and should begin with a unique package line. The package name is used in report generation and by the -p argument, so pick something mnemonic. The minimum needed for each package is the site, remote_dir and local_dir . On

finding a package line all the default values are reset.

If the package name is defaults then no site is contacted but the default values given for any keywords are changed. Personally I begin my config files with:

```
package=defaults
    remote_password=ukuug-soft@doc.ic.ac.uk
    get_newer=yes
    get_size_change=yes
```

If the package is not defaults then mirror will perform the

following steps. Unless an internal failure is detected any error will cause the current package to be skipped and the next one tried.

If Mirror is not already connected to the site it will disconnect from any site it is already connected to then attempt to connect to the remote site's ftp daemon. It will then login using the given remote username and password. Once connected mirror turns on binary mode transfers. Next it changes to the given local directory and scans it to get the details of the local files that already exist, if necessary the local directory will be created. Once this is completed the remote directory is similarly scanned. Mirror does this by changing to the remote directory and runing the ftp LIST command, passing the -lRt options. (I am not very happy about this bit and hope to allow it to pull back a file containing the remote directory listing instead.) Each remote pathname will have any specified mappings performed on it to create a local pathname. Then any checks specified by the exclude_patt, max_days, get_newer and get_size_change keywords are applied on names of files or symlinks. Only exclude_patt checking is applied to directories.

The above creates a list of all required remote files and the local path names to store them in.

Once the directory listing is completed all required files are fetched from the remote site into their local path names. This is done by pulling the file into a temporary file in the target directory. If required the temporary file is compressed. The temporary file is renamed when the transfer is successful.

EXAMPLES

Here is the mirror.defaults file from the archive on src.doc.ic.ac.uk.

```
# This is the defaul mirror settings used by my site:
# src.doc.ic.ac.uk (146.169.3.7)
# This is home of the UKUUG Software Distribution Service
```

```
#
# Lee McLoughlin <lmjm@doc.ic.ac.uk>

# Set my defaults
package=defaults
    # Keep all local_dirs relative to here
    local_dir=/vol/public/
    remote_password=ukuug-soft@doc.ic.ac.uk
    mail_to=lmjm
    dir_mode=0755
    file_mode=0444
    user=0
    group=0
    get_newer=yes
    get_size_change=yes
    # Don't overwrite my mirror log with the remote one.
    # Don't pull back any of their mirror temporary files.
    exclude_patt=^.mirror$|^MIRROR.LOG$|^.in..*.$|^#.*#|^lost+found/
    # Don't compress arc, zip, boo, readme files and index.txt files
    compress_excl+|.arc$|.zip$|.boo$|[Rr][Ee][Aa][Dd][Mm][Ee]|index.txt
    # Keep a log file in each updated directory
    update_log=.mirror
```

And here is part of the mirror.config:

```
package=gnu
    comment=Powerful and free Un*x utilities
    site=prep.ai.mit.edu
    remote_dir=/pub/gnu
    local_dir+gnu
    exclude_patt+|^ListArchives/|^lost+found/|^scheme-7.0/|^.history
    # I tend to only keep the lastest couple of versions of things
    # this stops mirror from re-pulling the older versions I've removed
    max_days=30

package=elisp-archive
    site=tut.cis.ohio-state.edu
    remote_dir=/pub/gnu/emacs/elisp-archive
    local_dir+gnu/EmacsBits/elisp-archive

package=X
    comment=The X Area at export
    site=export.lcs.mit.edu
    remote_dir=/contrib
    local_dir+X/contrib
    # go-1.0.b.tar.Z is immense so I store it split locally.
    exclude_patt+|^unicom|^go-1.0.b.tar.Z
```

```
        # I tend to only keep the lastest couple of versions of things
        # this stops mirror from re-pulling the older versions I've removed
        max_days=30

    package=cnews
        comment=The C News system
```



```
        site=ftp.cs.toronto.edu
        remote_dir=/pub/c-news
        local_dir+news/c
        compress_excl+|patches/PATCHDATES
        compress_patt=patches/
        exclude_patt+|^c-news.Z

    # and on, and on ...
```

HINTS

Always on adding in a new package check it out first turning on the -_n option.

If you are adding to an existing archive then it is ususally best to force the timestamps so time comparisions will work.

Try and have all packages that are being retrieved from the same site one after the other. That way mirror will only have to login once.

Keep your default settings in a separate file. That way you will, hopefully, be able to share mirror details with others.

NETIQUETTE

If you are going to mirror a remote site please obey any restrictions that the site administrators place on access. You can generally find the restrictions on connecting into the archive using the standard ftp command. Any restrictions are normally given as a login banner or in a, hope-

fully, obvious file.

Here are, what I hope are, some good general rules.

Only mirror a site well outside the working hours of both the local and remote sites.

It is probably unfriendly to try to mirror a remote site more than once a day.

Before trying to mirror a remote site try and find the packages you want from local archives, no one will be pleased if you soak up a lot of network bandwidth needlessly.

If you have a local archive then tell people about it so they don't have to waste bandwidth and CPU at the remote site.

Do remember to check your config files from time to time in case the remote archive has changed their access restrictions.

Check the remote site regularly for any new restrictions.

SEE ALSO

perl(l), ftp(1)

BUGS

The remaining keywords need to be implemented.

Should be able to mirror non Un*x sites (it may be able to but I have not tested this - the remote ls is the problem).

It should restart file transfers where they left off.

Hanging data transfers should be detected.

Should allow for multiple packages from the same host, efficiently.

Some of the netiquette guidelines should be enforced.

Should be able to cope with links as well as symlinks.

Beginning to suffer from _□c_□r_□e_□e_□p_□i_□n_□g _□f_□e_□a_□t_□u_□r_□i_□s_□m.

AUTHOR

Written by Lee McLoughlin <lmjm@doc.ic.ac.uk>. It uses the ftp.pl package by: Alan R. Martello <al@ee.pitt.edu> which uses the chat2.pl package by: Randal L. Schwartz <merlyn@iwarp.intel.com>

Article 5397 of comp.lang.perl:
Xref: feenix.metronet.com comp.infosystems.www:1336 comp.lang.perl:5397
Newsgroups: comp.infosystems.www,comp.lang.perl
Path:
feenix.metronet.com!news.ecn.bgu.edu!usenet.ins.cwru.edu!howland.reston.ans.net!xlink.net!scsing.switch.ch!news.unigenet
From: oscar@cui.unige.ch (Oscar Nierstrasz)
Subject: perl script for connecting to http servers
Message-ID: <1993Aug27.094503.15947@news.unige.ch>
Sender: usenet@news.unige.ch
Reply-To: oscar@cui.unige.ch
Organization: University of Geneva, Switzerland
Date: Fri, 27 Aug 1993 09:45:03 GMT
Lines: 73

Well, the problems I had in my previous posting apparently didn't have to
do with what I thought. (I still don't really understand what was wrong.)

Anyway, for what it is worth, here's a little perl script to grab
html pages given a list of URLs. It may be useful as a start to build
more elaborate scripts (like robots).

Oscar

```
#!/local/bin/perl -s
#
# hget                  --- get an html page from an http server
#
# Oscar Nierstrasz 26/8/93 oscar@cui.unige.ch
#
# following the example of Gene Spafford's ftpget

require "chat2.pl";

die "Usage: hget <http-url> ...\n" unless $#ARGV >= 0;
$timeout = 60;

foreach $url (@ARGV) {
        if ($url =~ m|^http://(.*)|) {
                $host = $1;
                $port = 80; # default
                $request = "/";         # default
                ($host =~ s|^([^/]+)/(.*)$|$1|) && ($request = $2);
                ($host =~ s/:(\d+)$//) && ($port = $1);
        }
        # relative URL, so assume previous host & port:
        elsif ($url =~ /^http:(.*)/) {
                $request = $1;
```

```
                unless ($host) {
                        warn "hget: no host for $url\n";
                        next;
                }
        }
        else { warn "hget: $url is not an http URL\n"; next; }
        &http_get($host,$port,$request);
}

sub http_get {
        local($host,$port,$request) = @_;
        ($handle = &chat'open_port($host, $port))
                || die "chat'open($host,$port): $!\n";
        &chat'print($handle,"GET $request\n")
                || die "chat'print(GET $request): $!\n";
        $done = 0;
        do{
                &chat'expect($handle, $timeout,
                        '.', q{print "$chat'thisbuf"},
                        'EOF', '$done = 1',
                        'TIMEOUT', 'print STDERR "TIMEOUT\n"; $done = 1'
                        )
                || die "chat'expect: $!\n";
        } until $done;
        &chat'close($handle);
}

__END__
```

Dr. O.M. Nierstrasz, Centre Universitaire d'Informatique
University of Geneva, 24, rue General-Dufour, CH-1211 Geneva 4, Switzerland
Tel: +41 22 705.7664 Secr: 705.7770 Fax: 320.2927
E-mail: oscar@cui.unige.ch Home: 733.9568

Article 3893 of comp.lang:perl:
Xref: feenix.metronet.com comp.lang.perl:3893
Path:
feenix.metronet.com!news.ecn.bgu.edu!mp.cs.niu.edu!ux1.cso.uiuc.edu!howland.reston.ans.net!gatech!asuvax!ncar!vexcper!aspen.craycos.com!scott
From: scott@craycos.com (Scott Bolte)
Newsgroups: comp.lang.perl
Subject: Re: ftpget script (provided)
Message-ID: <1993Jun30.175050.14781@craycos.com>
Date: 30 Jun 93 17:50:50 GMT
References: <C9FtMu.5zA@newsserver.technet.sg>
Organization: Cray Computer Corporation
Lines: 832

In article <C9FtMu.5zA@newsserver.technet.sg> mathias@solomon.technet.sg (Mathias Koerber) writes:

...

>What I'd like is a script that just goes to a specified host, logs in,
>cd's to the directory and then ftps the named files. Would be nice if it
>sat atop ftp.pl.
>
>Does anyone already have such a beast?

I wrote just such a beast a while ago and have been using it ever since. (In fact, it was used to pick up perl 4.036 just two nights ago.) It does not use ftp.pl, all it requires is ftp.

If you have the at command available it will even do the transfers at night to ease the load on the net.

Here is a brief description of the scripts shar'ed together below:

ftpget Obtains a single file from a remote host.

ftpmget Obtains a set of files from a remote host.

ftpls Run ls -l in a specific directory on a remote host.

ftplist Given a set of hosts & directories make a series of calls to ftpls. The results are placed in files that reflect the listing source.

Enjoy,

Scott

Scott Bolte scott@craycos.com +1 719 540 4186
Cray Computer Corporation, 1110 Bayfield Drive, Colorado Springs, CO 80906
As anyone here will tell you: I speak for myself.

*** On the Internet no one can hear you scream ***

-------- rip, cut, sever, slice, dice or tear here ----------

```
#! /bin/sh
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#                "End of shell archive."
# Contents:  README ftpget.pl ftplist.pl ftpls.pl ftpmget.pl
# Wrapped by scott@copper on Wed Jun 30 11:48:53 1993
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'README' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'README'\"
else
echo shar: Extracting \"'README'\" \(1371 characters\)
sed "s/^X//" >'README' <<'END_OF_FILE'
X                       Ftp Perl Scripts, Version 0.1
X
X
X                       Copyright (c) 1993, Scott Bolte
X
These scripts are free software. But if you want to send money anyway I
won't mind. The scripts are distributed in the hope that they will be
useful, but WITHOUT ANY WARRANTY.
X
You can take the scripts and use them yourself. You can even include
them in a product you sell as long as they do not add to the product
cost. But the copyright must be maintained.
X
The files in this directory were cobbled together in the last few
weeks. I have been using to automate ftp transfers. They also can be
asked to run at night in order to spread the demands on the net.
X
X       ftpget      Obtains a single file from a remote host.
X
X       ftpmget     Obtains a set of files from a remote host.
X
X       ftpls       Run ls -l in a specific directory on a remote
X                           host.
X
X       ftplist     Given a set of hosts & directories make a
X                           series of calls to ftpls. The results are
X                           placed in files that reflect the listing
X                           source.
```

```
X
X           ftpvaporware   A script, not yet written, that can look at the
X                                  data provided by ftplist. It will provide
X                                  notification that a item on an archive has
X                                  changed. Or that a new release of a program is
X                                  now available.
X
I would appreciate if any additions you make are sent to me. I will try
to incorporate them in any future releases.
X
X           Scott Bolte (scott@craycos.com)
X           1993.06.30
END_OF_FILE
if test 1371 -ne `wc -c <'README'`; then
   echo shar: \"'README'\" unpacked with wrong size!
fi
# end of 'README'
fi
if test -f 'ftpget.pl' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftpget.pl'\"
else
echo shar: Extracting \"'ftpget.pl'\" \(5666 characters\)
sed "s/^X//" >'ftpget.pl' <<'END_OF_FILE'
X#!/bin/perl
X
X##############################################################################
X#
X# Copyright 1993 Scott Bolte (scott@craycos.com)
X#
X#       Leave this copyright alone. But feel free to do with the script
X#       as you please. Sending me enhancements would be appreciated.
X#
X#       If you feel like pretending this is shareware, and want to
X#       send some money my way, feel free. I promise not to object.
X#
X# Summary of ftpget:
X#
X#           Obtain a file via ftp from a remote system. Anonymous ftp
X#           is used. Can be asked to delay the actual ftp request.
X#           On failure print out a call that can be used to try again.
X#
X# History:
X#
X#           1993.06.14 Initial implementation.
X#
X#           1993.06.15 Made remote and local file name processing safe
X#                                  even when the file name contains white space.
X#
X#           1993.06.28 Added interface to use "at" internally.
X#
```

```
X# Examples:
X#
X#          ftpget --at 23:30 prep.ai.mit.edu pub/gnu/perl-4.036.tar.gz
X#
X#          ftpget --at 23:30 prep.ai.mit.edu pub/gnu/perl-4.036.tar.gz new-perl
X#
X#   If you have an ftp hierarchy, as I do, the remote system can be
X#   derived from the current path.
X#
X#          cd ~/ftp/prep.ai.mit.edu
X#          ftpget --at 23:30 - pub/gnu/perl-4.036.tar.gz
X#
X###############################################################################
X
X$zero =  $0;
X$zero =~ s,.*/,,;
X
sub usage {
X           print <<EOS;
Usage: $zero [options] system remote_file [local_file]
X   --                                  Stop command line processing.
X   -a or --at time        At the given time, which should be in hh:mm format,
X                                       run the $0 command.
X   -d or --debug         Do not run the actual ftp command, use cat instead.
X   -v or --verbose       Enable the verbose status message.
X   -? or -h or --help    Print this usage statement.
X
X   If the system is "-" try to determine the remote system from
X   the current path.
XEOS
X}
X
X#
X# Set the default values.
X#
X$verbose  = 0;
X$debug    = 0;
X$cmd      = "ftp -n";    # change "ftp -n" to "cat -n" for testing.
X$time     = time;        # Get the time so we can recreate an "at"
X                                       # command if need be.
X
X#
X# Process the command line.
X#
while ($ARGV[0] =~ /^-./) {
X           $_ = shift;
X           if (/^--$/) {
X                       last;
X           }
X
```

```
X       if (/^-a$/ || /^--at$/) {
X               $delay_time = shift;
X               if ( $delay_time !~ /^\d\d?:\d\d$/ ) {
X                       print(STDERR
X                               "Bad time specification \"$delay_time\".\n");
X                       &usage();
X                       exit(1);
X               }
X               next;
X       }
X       if (/^-d$/ || /^--debug$/) {
X               $debug++;
X               $cmd   = "cat -n";
X               next;
X       }
X       if (/^-v$/ || /^--verbose$/) {
X               $verbose++;
X               next;
X       }
X
X       if (/^-\?$/ || /^-h$/ || /^--help$/) {
X               &usage();
X               exit(0);
X       }
X       print "I don't recognize this switch: $_\n";
X       &usage();
X       exit(1);
X}
X
X#
X# Verify we have the right number of positional arguments.
X#
if ( $#ARGV < 1 || $#ARGV > 2 ) {
X       &usage();
X       exit(1);
X}
X$system     = $ARGV[0];
X$remote_file = $ARGV[1];
X
X#
X# If the system spec was "-" try to figure out where we are. From that
X# we might be able to construct a default host.
X#
if ( $system eq "-" ) {
X       $system   = `/bin/pwd`;
X       $system   =~ s/\n//;
X       $original =  $system;
X       $system   =~ s,^.*/ftp/,,;
X       $system   =~ s,/.*$,,;
X       die("Could not determine system given path \"$original\".\n")
```

```
X                       if $system eq "";
X               print(STDERR "Derived system is \"$system\".\n") if $verbose;
X}
X
X#
X# Either take the optional third argument or construct it from the second.
X#
X
if ( "$remote_file" eq "" ) {
X               print(STDERR "Must specify a non-null file name.\n");
X               exit(1);
X}
if ( $#ARGV == 2 ) {
X               $local_file = $ARGV[2];
X               $explicit   = 1;
X} else {
X               $local_file =  $remote_file;
X               $local_file =~ s,/+$,,;
X               $local_file =~ s,.*/,,;
X               $explicit   =  0;
X
X               print(STDERR "Derived local path is \"$local_file\".\n") if $verbose;
X               if ( "$local_file" eq "" ) {
X                               print(STDERR "Unable to construct a local filename.\n");
X                               exit(1);
X               }
X}
X
X
X#
X# Make the path specifications safe even when they contain spaces.
X#
X$safe_remote =  $remote_file;
X$safe_local  =  $local_file;
X$safe_remote =~ s/(.*)/"$1"/ if $safe_remote =~ /\s/;
X$safe_local  =~ s/(.*)/"$1"/ if $safe_local  =~ /\s/;
X
X#
X# If a delay was asked for run the command later.
X#
if ( $delay_time ne "" ) {
X               $me  =  $0;
X               $me .= " --verbose"                     if $verbose;
X               $me .= " --debug"                       if $debug;
X               $me .= " $system $safe_remote";
X               $me .= " $safe_local"                   if $explicit;
X               $at  = "at $delay_time";
X               print(STDERR "At $delay_time the following command will be run:\n");
X               print(STDERR "  $me\n");
X               if ( $debug ) {
```

```
X                       print(STDERR "Skipping command in debug mode.\n");
X                       exit(0);
X               }
X               open(CMD, "|$at") || die("Could not run command ($at). $!\n");
X               print(CMD $me);
X               close(CMD);
X               exit(0);
X}
X
X#
X# Build the batch ftp command.
X#
X$user       = (getpwuid($<))[0];   # safe when run from "at".
X$localhost  = `hostname`; chop($localhost);
if ( $localhost !~ /\./ ) {
X               #
X               # If the host name does not have '.' notation try to
X               # get an alias. We then hope it is in domain name notation.
X               #
X               @fullhost  = gethostbyname($localhost);
X               $localhost = $fullhost[1] if $fullhost[1] ne "";
X}
X$template =
X"               open $system
X               user anonymous ${zero}4$user@$localhost
X               bin
X               get $safe_remote $safe_local
X               bye
X";
X
X#
X# Run the batch ftp command.
X#
print(STDERR "Running command ($cmd).\n") if $verbose;
open(CMD, "|$cmd") || die("Could not start command ($cmd). $!\n");
print CMD $template;
close(CMD);
X
X
X# Note whether or not the local file was obtained.
X#
if ( -f $local_file ) {
X               print("Obtained \"$local_file\" from $system.\n");
X} else {
X               $pwd=`pwd`;
X               chop($pwd);
X               @time = localtime($time);
X               $next = sprintf("%2d:%02d", $time[2], $time[1]);
X
X               $sep  = "\t\\\n\t    ";
```

```
X       $cmd = "$0";
X       $cmd .= "$sep--at $next";
X       $cmd .= "$sep$system";
X       $cmd .= "$sep$safe_remote";
X       $cmd .= "$sep$safe_local"       if $explicit;
X
X       print("
Unable to obtained \"$local_file\" from $system.
To try again the following command might be used:
X
cd $pwd;
X$cmd
X");
X}
END_OF_FILE
if test 5666 -ne `wc -c <'ftpget.pl'`; then
   echo shar: \"'ftpget.pl'\" unpacked with wrong size!
fi
chmod +x 'ftpget.pl'
# end of 'ftpget.pl'
fi
if test -f 'ftplist.pl' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftplist.pl'\"
else
echo shar: Extracting \"'ftplist.pl'\" \(1543 characters\)
sed "s/^X//" >'ftplist.pl' <<'END_OF_FILE'
X#!/bin/perl
X
X##############################################################################
X#
X# Copyright 1993 Scott Bolte (scott@craycos.com)
X#
X#       Leave this copyright alone. But feel free to do with the script
X#       as you please. Sending me enhancements would be appreciated.
X#
X#       If you feel like pretending this is shareware, and want to
X#       send some money my way, feel free. I promise not to object.
X#
X# Summary of ftplist:
X#
X#           Run ftpls on a bunch of systems. The results are put in files
X#           whose names map to the system/directory pair. Older copies are
X#           renamed before the new edition is obtained.
X#
X#           The expectation is that additional scripts will be run after
X#           this one.  They will compare the old and new listings to note
X#           changes.
X#
X# History:
X#
```

```
X#  -  1993.06.28     Initial implementation.
X#
X# Examples:
X#
X#      ftplist
X#
X##############################################################################
X
X%set = (
X           "agate.berkeley.edu",           "pub/386BSD/386bsd-0.1/unofficial",
X           "bsd.coe.montana.edu",          "pub/patch-kit",
X           "hrd769.brooks.af.mil",         "pub/FAQ",
X           "prep.ai.mit.edu",              "pub/gnu",
X           );
X
foreach $system (sort(keys(%set))) {
X           $file =  "$system:$set{$system}";
X           $file =~ s,/,_,g;
X           $old  = "$file.OLD";
X           if ( -f $file ) {
X                       $error  = "Could not rename \"$file\" to \"$old\". $!\n";
X                       $error .= "New listing of $set{$system} will not be obtained.\n";
X                       rename($file, $old)       || (warn($error), next);
X           }
X           $cmd = "ftpls $system $set{$system} > $file";
X           system($cmd) && die("Could not run command ($cmd). $!\n");
X}
END_OF_FILE
if test 1543 -ne `wc -c <'ftplist.pl'`; then
   echo shar: \"'ftplist.pl'\" unpacked with wrong size!
fi
# end of 'ftplist.pl'
fi
if test -f 'ftpls.pl' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftpls.pl'\"
else
echo shar: Extracting \"'ftpls.pl'\" \(1853 characters\)
sed "s/^X//" >'ftpls.pl' <<'END_OF_FILE'
X#!/bin/perl
X
X##############################################################################
X#
X# Copyright 1993 Scott Bolte (scott@craycos.com)
X#
X#       Leave this copyright alone. But feel free to do with the script
X#       as you please. Sending me enhancements would be appreciated.
X#
X#       If you feel like pretending this is shareware, and want to
X#       send some money my way, feel free. I promise not to object.
X#
```

```
X# Summary of ftpls:
X#
X#          Do a ls in a specific directory on a remote ftp system.
X#          Anonymous ftp is used.
X#
X# History:
X#
X#      1993.06.28      Initial implementation.
X#
X# Examples:
X#
X#      ftpls remote_system remote_dir
X#
X###############################################################################
X
X$cmd   = "ftp -n";       # change "ftp -n" to "cat -n" for testing.
X$zero =  $0;
X$zero =~ s,.*/,,;
X
X###############################################################################
X#
X# Verify the arguments
X#
if ( $#ARGV != 1 ) {
X           print(STDERR "Usage: $zero system remote_dir\n");
X           exit(1);
X}
X
X$system     = $ARGV[0];
X$remote_dir  = $ARGV[1];
X
X###############################################################################
X#
X# Build the batch ftp command.
X#
X$user       = (getpwuid($<))[0];   # safe when run from "at".
X$localhost  = `hostname`; chop($localhost);
if ( $localhost !~ /\./ ) {
X           #
X           # If the host name does not have '.' notation try to
X           # get an alias. We then hope it is in domain name notation.
X           #
X           @fullhost  = gethostbyname($localhost);
X           $localhost = $fullhost[1] if $fullhost[1] ne "";
X}
X$template =
X"           open $system
X           user anonymous ${zero}4$user@$localhost
X           bin
X           cd $remote_dir
```

```
X       ls -l
X       bye
X";
X
X##############################################################################
X#
X# Run the batch ftp command.
X#
open(CMD, "|$cmd") || die("Could not start command ($cmd). $!\n");
print CMD $template;
close(CMD);
X
exit(0);
END_OF_FILE
if test 1853 -ne `wc -c <'ftpls.pl'`; then
    echo shar: \"'ftpls.pl'\" unpacked with wrong size!
fi
chmod +x 'ftpls.pl'
# end of 'ftpls.pl'
fi
if test -f 'ftpmget.pl' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftpmget.pl'\"
else
echo shar: Extracting \"'ftpmget.pl'\" \(7128 characters\)
sed "s/^X//" >'ftpmget.pl' <<'END_OF_FILE'
X#!/bin/perl
X
X##############################################################################
X#
X# Copyright 1993 Scott Bolte (scott@craycos.com)
X#
X#       Leave this copyright alone. But feel free to do with the script
X#       as you please. Sending me enhancements would be appreciated.
X#
X#       If you feel like pretending this is shareware, and want to
X#       send some money my way, feel free. I promise not to object.
X#
X# Summary of ftpmget:
X#
X#       Obtain a set of files via ftp from a remote system. Anonymous
X#       ftp is used. Can be asked to delay the command until a later
X#       time. Such a request will result in "at" being used.
X#
X# History:
X#
X#       1993.06.14 Initial implementation.
X#
X#       1993.06.15 Changed so that if a file contains white space
X#                               it is obtained with get instead of mget.
X#
```

```
X#          1993.06.30 Added --at option to allow delayed operation.
X#
X# Examples:
X#
X#          ftpmget hrd769.brooks.af.mil pub/FAQ FAQ_07 FAQ_09
X#
X#          cd ~/ftp/hrd769.brooks.af.mil
X#          ftpmget - pub/FAQ FAQ_07 FAQ_09
X#
X#          ftpmget hrd769.brooks.af.mil pub/FAQ - << EndOfList
X#                      FAQ_07
X#                      FAQ_09
X#          EndOfList
X#
X###############################################################################
X
X$zero =  $0;
X$zero =~ s,.*/,,;
X
sub usage {
X
X       print <<EOS;
Usage: $zero [options] system remote_dir file1 [... fileN]
X or
X     $zero [options] system remote_dir -
X
X   --                          Stop command line processing.
X   -a or --at time         At the given time, which should be in hh:mm format,
X                               run the $0 command.
X   -d or --debug          Do not run the actual ftp command, use cat instead.
X   -v or --verbose        Enable the verbose status message.
X   -? or -h or --help     Print this usage statement.
X
X   If the system is "-" try to determine the remote system from
X   the current path.
X
X   If a "-" is given instead of a list of files the list is read
X   from standard input.
XEOS
X}
X
X###############################################################################
X#
X#          Set the default values.
X#
X
X$verbose  = 0;
X$debug    = 0;
X$cmd      = "ftp -n"; # change "ftp -n" to "cat -n" for testing.
X$time     = time;     # Get the time so we can recreate an "at"
```

```
X                               # command if need be.
X$pwd     =`/bin/pwd`; chop($pwd);
X
X##############################################################################
X#
X#       Process the command line.
X#
X
while ($ARGV[0] =~ /^-./) {
X       $_ = shift;
X       if (/^--$/) {
X               last;
X       }
X
X       if (/^-a$/ || /^--at$/) {
X               $delay_time = shift;
X               if ( $delay_time !~ /^\d\d?:\d\d$/ ) {
X                       print(STDERR
X                               "Bad time specification \"$delay_time\".\n");
X                       &usage();
X                       exit(1);
X               }
X               next;
X       }
X       if (/^-d$/ || /^--debug$/) {
X               $debug++;
X               $cmd   = "cat -n";
X               next;
X       }
X       if (/^-v$/ || /^--verbose$/) {
X               $verbose++;
X               next;
X       }
X
X       if (/^-\?$/ || /^-h$/ || /^--help$/) {
X               &usage();
X               exit(0);
X       }
X       print "I don't recognize this switch: $_\n";
X       &usage();
X       exit(1);
X}
X
X##############################################################################
X#
X# Verify the positional arguments
X#
if ( $#ARGV < 2 ) {
X       &usage();
X       exit(1);
```

```
X}
X
X$system       = $ARGV[0];
X$remote_dir  = $ARGV[1];
X
X#############################################################################
X#
X# If the system spec was "-" try to figure out where we are. From that
X# we might be able to construct a default host.
X#
if ( $system eq "-" ) {
X           $system   = `/bin/pwd`;
X           $system   =~ s/\n//;
X           $original =  $system;
X           $system   =~ s,^.*/ftp/,,;
X           $system   =~ s,/.*$,,;
X           die("Could not determine system given path \"$original\".\n")
X                       if $system eq "";
X           print(STDERR "Derived system is \"$system\".\n") if $verbose;
X}
X
X#############################################################################
X#
X# Get the list of requested files. Either take the files from
X# the command line or from stdin.
X#
if ( $ARGV[2] ne "-" ) {
X           for( $argc = 2; $argc <= $#ARGV; $argc++) {
X                       $file =         $ARGV[$argc];
X                       push(@files, $file);
X           }
X} else {
X           while(<STDIN>) {
X                       chop;
X                       $file =         $_;
X                       $file =~ s/^\s+//;                  # leading white space
X                       $file =~ s/\s+$//;                  # trailing white space
X                       $file =~ s/^"(.*)"$/$1/;  # Remove enclosing quotes
X                       push(@files, $file);
X           }
X}
X
X#
X# Make spaces safe for all mankind. N. Armstrong.
X#
foreach $file (@files) {
X           next if $file eq "";                                # skip empty names
X           $file =~ s/(.*)/"$1"/ if $file =~ /\s/; # add quotes if need be.
X           push(@tmp, $file);
X}
```

```
X@files = @tmp;
undef       @tmp;
X
X###############################################################################
X#
X#       If a delay was asked for run the command later.
X#
X
if ( $delay_time ne "" ) {
X       $me =       $0;
X       $me .= " --verbose"                 if $verbose;
X       $me .= " --debug"                   if $debug;
X       $me .= " $system $remote_dir - << End_Of_List\n";
X       foreach $file (@files) {
X               $me .= "\t$file\n";
X       }
X       $me .= "End_Of_List\n";
X
X       $at  = "at $delay_time";
X
X       print(STDERR "At $delay_time the following command will be run:\n");
X       print(STDERR "  $me\n");
X       if ( $debug ) {
X               print(STDERR "Skipping command in debug mode.\n");
X               exit(0);
X       }
X       open(CMD, "|$at") || die("Could not run command ($at). $!\n");
X       print(CMD $me);
X       close(CMD);
X       exit(0);
X}
X
X###############################################################################
X#
X#       Build the batch ftp command.
X#
X
X$user       = (getpwuid($<))[0]; # safe when run from "at".
X$localhost = `hostname`;
X           chop($localhost);
if ( $localhost !~ /\./ ) {
X       #
X       # If the host name does not have '.' notation try to
X       # get an alias. We then hope it is in domain name notation.
X       #
X       @fullhost  = gethostbyname($localhost);
X       $localhost = $fullhost[1] if $fullhost[1] ne "";
X}
X$template =
X"   open $system
```

```
X   user anonymous ${zero}4$user@$localhost
X   bin
X   prompt
X   cd $remote_dir
X";
foreach $file (@files) {
X           if ( $file =~ /\s/ ) {
X                       push(@space_files, $file);
X                       next;
X           }
X           if ( length($line) + length($file) + 1 > 75) {
X                       $template .= sprintf("$line\n");
X                       $line = "";
X           }
X           $line  = "    mget" if $line eq "";
X           $line .= " $file";
X}
X$template .= "$line\n";
foreach $file (@space_files) {
X           $template .= "                get $file\n";
X}
X$template .= "$bye\n";
X
X#############################################################################
X#
X#           Run the batch ftp command.
X#
X
open(CMD, "|$cmd") || die("Could not start command ($cmd). $!\n");
print CMD $template;
close(CMD);
X
X#############################################################################
X#
X#           Report whether or not the files were obtained.
X#
X
print("\nReport for file transfers from $system.\n");
print("  Remote directory \"$remote_dir\".\n");
print("  Local directory \"$pwd\".\n");
foreach $file (@files) {
X           if ( -f $file ) {
X                       print("    Obtained \"$file\".\n");
X                       next;
X           }
X
X           if ( $file =~ /^".*"$/ ) {
X                       $file =~ s/^"(.*)"$/$1/;
X                       if ( -f $file ) {
X                                   print("    Obtained \"$file\".\n");
```

```
X                               next;
X                       }
X               }
X               push(@again, $file);
X}
X
X###############################################################################
X#
X#               If there was a problem getting all the files print out a command
X#               that can try again later.
X#
X
if ( $#again >= $[ ) {
X               @time = localtime($time);
X               $next = sprintf("%2d:%02d", $time[2], $time[1]);
X
X               print <<EOS;
X
Unable to obtained some files from $system.
To try again the following command might be used:
X
cd $pwd
X$0 --at $next $system $remote_dir - << End_Of_List
XEOS
X
X               foreach $file (@again) {
X                               print("               $file\n");
X               }
X               print("End_Of_List\n");
X}
END_OF_FILE
if test 7128 -ne `wc -c <'ftpmget.pl'`; then
   echo shar: \"'ftpmget.pl'\" unpacked with wrong size!
fi
chmod +x 'ftpmget.pl'
# end of 'ftpmget.pl'
fi
echo shar: End of shell archive.
exit 0
--
```

Scott Bolte scott@craycos.com +1 719 540 4186
Cray Computer Corporation, 1110 Bayfield Drive, Colorado Springs, CO 80906

As anyone here will tell you: I speak for myself.

*** On the Internet no one can hear you scream ***

```
##! /bin/sh
#From cs.utexas.edu!asuvax!gatech!news.byu.edu!eff!iWarp.intel.com|news Fri May  1 14:52:11 CDT 1992
#Article: 10383 of comp.lang.perl
#Path: cse.uta.edu!cs.utexas.edu!asuvax!gatech!news.byu.edu!eff!iWarp.intel.com|news
#From: merlyn@iWarp.intel.com (Randal L. Schwartz)
#Newsgroups: comp.lang.perl
#Subject: Re: Perl FTP Interface (Need Example) (Do I Use expect.pl?)
#Message-ID: <1992May1.152710.22905@iWarp.intel.com>
#Date: 1 May 92 15:27:10 GMT
#References: <3604@ucru2.ucr.edu> <18604@ector.cs.purdue.edu>
#Sender: news@iWarp.intel.com
#Reply-To: merlyn@iWarp.intel.com (Randal L. Schwartz)
#Organization: Stonehenge; netaccess via Intel, Beaverton, Oregon, USA
#Lines: 277
#In-Reply-To: spaf@cs.purdue.EDU (Gene Spafford)
#Nntp-Posting-Host: v.iwarp.intel.com
#
#In article <18604@ector.cs.purdue.edu>, spaf@cs (Gene Spafford) writes:
#| Well, I guess now is as good a time as any.
#|
#| I have put together a "ftp library package" that allows one to
#| construct fun little ftp programs.  It works well for me -- I've built
#| a mirroring program and a couple of command-line ftp commands.
#|
#| None of this is documented (I got really busy just when I finished
#| testing this).  I'll include the code for the library here, and
#| the code for my two example commands.  One command lets you "ls" a
#| remote directory using ftp, and the other lets you get arbitrary
#| files, in either binary or ascii mode.  I'm half done with one that
#| will let you fetch a remote tree, ala "rcp -r"
#
#Well, hey, since I have a little script that does kinda the same thing
#(the one you're "half done" with), I'll post it.  Amazingly enough,
#it *also* uses chat2.pl :-).
#
#It presumes a BSD-like remote host, and fails miserably on any unusual
#forms of ftpd.  Try it first to see, though.
# This is a shell archive.  Remove anything before this line, then unpack
# it by saving it into a file and typing "sh file".  To overwrite existing
# files, type "sh file -c".  You can also feed this as standard input via
# unshar, or by typing "sh <file", e.g..  If this archive is complete, you
# will see the following message at the end:
#                "End of shell archive."
# Contents:  ftpr
# Wrapped by merlyn@iwarpv on Fri May  1 08:22:59 1992
PATH=/bin:/usr/bin:/usr/ucb ; export PATH
if test -f 'ftpr' -a "${1}" != "-c" ; then
  echo shar: Will not clobber existing file \"'ftpr'\"
else
echo shar: Extracting \"'ftpr'\" \(5037 characters\)
```

```
sed "s/^X//" >'ftpr' <<'END_OF_FILE'
X#!/usr/bin/perl
X
X## ftpr, last update 91/08/16
X## usage: ftpr [-a] [-d] [-t timeout] [-n] hostname topdir yes-regex except-regex
X## topdir may be whitespace-separated list of topdirs
X## yes-regex defaults to . (meaning everything)
X## except-regex defaults to ' ' (meaning no exceptions)
X
Xpush(@INC, '/local/merlyn/lib/perl');
X
Xrequire 'chat2.pl';
X
X$| = 1; # not much output, but we like to see it as it happens
X$timeout = 60;
X$dasha = "";
X$nflag = 0;
X$host = "localhost";
X$topdir = ".";
X$yesregex = ".";
X$noregex = " ";
X$user = "anonymous";
X$pass = 'merlyn@iwarp.intel.com';
X
X{
X    last unless $ARGV[0] =~ /^-/;
X    $_ = shift;
X    $trace++, redo if /^-d/; # debug mode
X    $timeout = $1, redo if /^-t(\d+)/;
X    $timeout = shift, redo if /^-t/;
X    $dasha = "-a", redo if /^-a/;
X    $nflag++, redo if /^-n/;
X    die "bad flag: $_";
X}
X
X$host = shift if @ARGV;
X$topdir = shift if @ARGV;
X$yesregex = shift if @ARGV;
X$noregex = shift if @ARGV;
X
Xdie "extra args: @ARGV" if @ARGV;
X
X($Control = &chat'open_port($host,21)) || die "open control: $!";
Xdie "expected 2dd for initial banner, got $_"
X        unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X&ctalk("user $user\n");
X$_ = &clisten($timeout);
Xunless (/^2\d\d/) { # might be logged in already:
X    die "expected 3dd for password query, got $_"
X        unless /^3\d\d/;
```

```
X           &ctalk("pass $pass\n");
X           die "expected 2dd for logged in, got $_"
X                   unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X}
X## all set up for a conversation
X
X@list = split(/\s+/,$topdir);
Xwhile ($dir = shift list) {
X           next if $seen{$dir}++;
X           print "listing $dir\n";
X           for (&list($dir)) {
X                   (warn "can't parse $_ in $dir"), next
X                           unless ($tag,$file) = /^(.).*\s(\S+)\s*$/;
X                   push(@list, "$dir/$file") if
X                           ($tag eq 'd') && ($file !~ /^\.\.?$/);
X                   if (            ($tag eq '-') &&
X                           ("$dir/$file" =~ /$yesregex/o) &&
X                           ("$dir/$file" !~ /$noregex/o) &&
X                           (! -e "$dir/$file")
X                   ) {
X                           print "fetching $dir/$file...\n";
X                           &get("$dir/$file","$dir/$file") unless $nflag;
X                   }
X           }
X}
X
X## shutdown
X&ctalk("quit\n");
X&clisten(5); # for trace
X&chat'close($Control);
Xexit(0);
X
Xsub ctalk {
X           local($text) = @_;
X           print "{$text}" if $trace;
X           &chat'print($Control,$text);
X}
X
Xsub clisten {
X           local($secs) = @_;
X           local($return,$tmp);
X           while (1) {
X                   $tmp = &chat'expect($Control, $secs, '(.*)\r?\n', '"$1\n"');
X                   print $tmp if $trace;
X                   $return .= $tmp;
X                   return $return if !length($tmp) || $tmp =~ /^\d\d\d /;
X           }
X}
X
Xsub dopen {
```

```
X           local($_);
X
X           local(@ret) = &chat'open_listen();
X           &ctalk("port " .
X                       join(",", @ret[0,1,2,3], int($ret[4]/256), $ret[4]%256) .
X                       "\n");
X           die "expected 2dd for data open, got $_"
X                       unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X           $Data = $ret[5];
X}
X
X<<'END_NOT_USED';
Xsub dtalk {
X           local($text) = @_;
X           print "{D:$text}" if $trace;
X           &chat'print($Data,$text);
X}
XEND_NOT_USED
X
Xsub dlisten {
X           local($secs,$forcereturn) = @_;
X           local($return,$tmp);
X           while (1) {
X                       $tmp = &chat'expect($Data, $secs,
X                                   '(.|\n)+', '$&',
X                                   TIMEOUT, '""',
X                                   EOF, 'undef');
X                       if (defined $tmp) {
X                                   print "[D:$tmp]" if $trace > 1;
X                                   $return .= $tmp;
X                                   return $return unless (!$forcereturn) && (length $tmp);
X                                               # if timeout, return what you have
X                       } else { # eof
X                                   return $return;
X                                               # maybe undef
X                       }
X           }
X}
X
Xsub dclose {
X           &chat'close($Data);
X}
X
X<<'END_NOT_USED';
Xsub nlst {
X           local($dir) = @_;
X           local(@files);
X           local($_,$tmp);
X
X           &dopen();
```

```
X           &ctalk("nlst $dasha $dir/.\n");
X           die "expected 1dd for nlst, got $_"
X                       unless ($_ = &clisten($timeout)) =~ /^1\d\d/;
X           $_ = "";
X           while (1) {
X                       $tmp = &dlisten($timeout);
X                       last unless defined $tmp;
X                       $_ .= $tmp;
X           }
X           @files = sort grep(!/^\.\.?$/, split(/\r?\n/))
X                       unless /^ls: /;
X           die "expected 2dd for nlst complete, got $_"
X                       unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X           &dclose();
X           @files;
X}
XEND_NOT_USED
X
Xsub list {
X           local($dir) = @_;
X           local(@files);
X           local($_,$tmp);
X
X           &dopen();
X           &ctalk("list $dasha $dir/.\n");
X           die "expected 1dd for list, got $_"
X                       unless ($_ = &clisten($timeout)) =~ /^(.*\n)*1/;
X           $_ = "";
X           while (1) {
X                       $tmp = &dlisten($timeout);
X                       last unless defined $tmp;
X                       $_ .= $tmp;
X           }
X           @files = grep(/^\S[rwx\-]{8}/, split(/\r?\n/));
X           die "expected 2dd for list complete, got $_"
X                       unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X           &dclose();
X           @files;
X}
X
Xsub get {
X           local($from, $to) = @_;
X           local($todir,*OUT);
X
X           ($todir = "./$to") =~ s#(.*)/.*#$1#;
X           system "mkdir -p $todir" unless -d $todir;
X           (warn "cannot create $to.TMP: $!"), return
X                       unless open(OUT, ">$to.TMP");
X           select((select(OUT),$|=1)[0]);
X           &ctalk("type i\n");
```

```
X           die "expected 2dd for type i ok, got $_"
X                   unless ($_ = &clisten($timeout)) =~ /^2\d\d/;
X           &dopen();
X           &ctalk("retr $from\n");
X           unless (($_ = &clisten($timeout)) =~ /^1\d\d/) {
X                   warn "expected 1dd for retr, got $_";
X                   close(OUT);
X                   unlink("$to.TMP");
X                   &dclose();
X                   return;
X           }
X           {
X                   $_ = &dlisten($timeout,1);
X                   last unless defined $_;
X                   print OUT;
X                   redo;
X           }
X           close(OUT);
X           unless (($_ = &clisten($timeout)) =~ /^2\d\d/) {
X                   warn "expected 2dd for retr complete, got $_";
X                   close(OUT);
X                   unlink("$to.TMP");
X                   &dclose();
X                   return;
X           }
X           &dclose();
X           rename("$to.TMP","$to") || warn "cannot rename $to.TMP to $to: $!";
X}
END_OF_FILE
if test 5037 -ne `wc -c <'ftpr'`; then
    echo shar: \"'ftpr'\" unpacked with wrong size!
fi
chmod +x 'ftpr'
# end of 'ftpr'
fi
echo shar: End of shell archive.
exit 0
--
/=Randal L. Schwartz, Stonehenge Consulting Services (503)777-0095 ==========\
| on contract to Intel's iWarp project, Beaverton, Oregon, USA, Sol III      |
| merlyn@iwarp.intel.com ...!any-MX-mailer-like-uunet!iwarp.intel.com!merlyn |
\=Cute Quote: "Intel: putting the 'backward' in 'backward compatible'..."====/
```

CLIPPEDIMAGE= JP360229138A
PAT-NO: JP360229138A
DOCUMENT-IDENTIFIER: JP 60229138 A
TITLE: INFORMATION **RETRIEVING** SERVICE SYSTEM

PUBN-DATE: November 14, 1985

INVENTOR-INFORMATION:
NAME
KIKUCHI, SHIRO
FUKUHARA, YOSHIZO
AKIYAMA, TAKASHI
HATAKEYAMA, KOZO
KATO, YUICHI

ASSIGNEE-INFORMATION:

NAME	COUNTRY
NIPPON TELEGR & TELEPH CORP <NTT>	N/A
NEC CORP	N/A
HITACHI LTD	N/A
FUJITSU LTD	N/A

APPL-NO: JP59084388
APPL-DATE: April 26, 1984

INT-CL_(IPC): G06F007/28; G06F015/40

ABSTRACT:
PURPOSE: To inform instantaneously or at a designated time the retieval information to a requester via a communication line by registering previously the telephone number of a receiver, an instantaneous/time point designation flag, a transmission time point, etc. to a **retrieval** equation register memory in addition to a **retrieval** equation.

CONSTITUTION: The new information supplied from an input device 11 is stored to a desired address on a data base 14 via a **retrieval** processor 13. At the same time, the processor 13 checks the coincidence of the input information with all **retrieval** equations registered to a **retrieval** equation register memory 12. When the coincidence of input information is obtained, the telephone number of a receiver, an instantaneous/time point designation flag, and a transmission time point corresponding to the relevant **retrieval** equation are read out of the memory 12 and then stored in an output memory 16 with addition of the head address of the input information and the data length. While a cycle reading circuit 17 scans a memory 16 **periodically** . The desired information is read out to a data transmitter 15 in case the transmission time point is coincident with an instantaneous output or a present time point. A line to a terminal device 20 designated for the receiver by the receiver telephone number through a communication network 19. Then the information is transmitted.

CLIPPEDIMAGE= JP403230234A
PAT-NO: JP403230234A
DOCUMENT-IDENTIFIER: JP 03230234 A
TITLE: **RETRIEVING** METHOD FOR DATA BASE

PUBN-DATE: October 14, 1991

INVENTOR-INFORMATION:
NAME
NISHIDA, MASATOSHI
FUJIWARA, KEIKO

ASSIGNEE-INFORMATION:

NAME	COUNTRY
OKI SOFTWARE KK	N/A
OKI ELECTRIC IND CO LTD	N/A

APPL-NO: JP02025158
APPL-DATE: February 6, 1990

INT-CL_(IPC): G06F012/00; G06F015/40

ABSTRACT:
PURPOSE: To speed up **retrieval** operation by providing a terminal device with a local file and a terminal controller with an update history file.

CONSTITUTION: The terminal controller 30 which repeats data between a center 10 and the terminal device 40 is provided with the update history file 31 consisting of only data which are updated newly when a master file 11 is updat ed. The terminal device 40, on the other hand, is provided with the local file 41 copying the master file 11 **periodically** . The latest data base can, therefore, be relrieved by referring to the local file 41 and update history file 31 even unless the local file 41 and master file 1 are updated at the same time. Conse quently, the data base can be **retrieved** more efficiently and faster.

CLIPPEDIMAGE= JP404142620A
PAT-NO: JP404142620A
DOCUMENT-IDENTIFIER: JP 04142620 A
TITLE: BATH INQUIRY SYSTEM

PUBN-DATE: May 15, 1992

INVENTOR-INFORMATION:
NAME
SHINOZAKI, YUKIHIRO

ASSIGNEE-INFORMATION:
NAME COUNTRY
NEC CORP N/A

APPL-NO: JP02265758
APPL-DATE: October 3, 1990

INT-CL_(IPC): G06F009/06

ABSTRACT:
PURPOSE: To protect generation of erroneous input by **retrieving** plural master files in accordance with a program obtained based on an inquiry condition and outputting the content of the **retrieved** master files to a predetermined document.

CONSTITUTION: To the batch inquiry system is provided with an inquiry condition control file for controlling an inquiry condition and plural master files for storing the content subject to **retrieval** in advance. Further, in accordance with a program obtained based on the inquiry condition, plural master files are **retrieved** and the output to a predetermined document. Accordingly, by controlling plural inquiry conditions using a file, plural inquiries can be continuously executed at a time. With this, plural inquiries that one wants to process **periodically** can be executed without carrying out burdensome input of inquiry conditions for every inquiry and erroneous input of inquiry conditions can be eliminated.

CLIPPEDIMAGE= JP362053085A
PAT-NO: JP362053085A
DOCUMENT-IDENTIFIER: JP 62053085 A
TITLE: VIDEO TEX TERMINAL EQUIPMENT FOR USER

PUBN-DATE: March 7, 1987

INVENTOR-INFORMATION:
NAME
EDAMOTO, MASANORI

ASSIGNEE-INFORMATION:

NAME	COUNTRY
NEC CORP	N/A

APPL-NO: JP60194303
APPL-DATE: September 2, 1985

INT-CL_(IPC): H04N007/173; G06F013/00 ; H04M011/00

ABSTRACT:
PURPOSE: To automatically display a special picture with a prescribed time interval by providing a circuit to store a picture-retrieving state and a circuit to automatically request for a picture to the video TEX center **periodically** according to the information in the said circuit.

CONSTITUTION: A **periodic** time interval is set in a time-setting command circuit 19 through a key pad 16. The circuit 19 instructs an automatic picture-request control circuit 18 to start to **automatic picture retrieval** operation at every time after the lapsing of the set time period. after being instructed, the circuit 18 obtains a desired picture from the video TEX center is accordance with the stored information which is a storage of the retrieving process after connecting with the video TEX center in a picture retrieval storage circuit 17. Until the **periodic** time interval is released through the key pad 16, the above operation is repeated, and the user can observe the latest information of a desired picture in a displayer part 15.

Network Working Group
Request for Comments: 1068

A. DeSchon
R. Braden
ISI

August 1988

Background File Transfer Program (BFTP)

Status of This Memo

This memo describes an Internet background file transfer service that is built upon the third-party transfer model of FTP. No new protocols are involved. The purpose of this memo is to stimulate discussion on new Internet service modes. Distribution of this memo is unlimited.

1. Introduction

For a variety of reasons, file transfer in the Internet has generally been implemented as an interactive or "foreground" service. That is, a user runs the appropriate local FTP user interface program as an interactive command and requests a file transfer to occur in real time. If the transfer should fail to complete for any reason, the user must reissue the transfer request. Foreground file transfer is relatively simple to implement -- no subtleties of queuing or stable storage -- and in the early days of networking it provided excellent service, because the Internet/ARPANET was lightly loaded and reasonably reliable.

More recently, the Internet has become increasingly subject to congestion and long delays, particularly during times of peak usage. In addition, as more of the world becomes interconnected, planned and unplanned outages of hosts, gateways, and networks sometimes make it difficult for users to successfully transfer files in foreground.

Performing file transfer asynchronously (i.e., in "background"), provides a solution to some of these problems, by eliminating the requirement for a human user to be directly involved at the time that a file transfer takes place. A background file transfer service

requires two components: a user interface program to collect the parameters describing the required transfer(s), and a file transfer control (FTC) daemon to carry them out.

Background file transfer has a number of potential advantages for a user:

o No Waiting

The user can request a large transfer and ignore it until a notification message arrives through some common channel (e.g., electronic mail).

o End-to-end Reliability

The FTC daemon can try a transfer repeatedly until it either succeeds or fails permanently. This provides reliable end-to-end delivery of a file, in spite of the source or destination host being down or poor Internet connectivity during some time period.

o Multiple File Delivery

In order for background file transfer to be accepted in the Internet, it may have to include some "value-added" services. One such service would be an implementation of a multiple file transfer capability for all hosts. Such a facility is suggested in RFC-959 (see the description of "NLST") and implemented in some User-FTP programs.

o Deferred Delivery

The user may wish to defer a large transfer until an off-peak period. This may become important when parts of the Internet adopt accounting and traffic-based cost-recovery mechanisms.

There is a serious human-engineering problem with background file transfer: if the user makes a mistake in entering parameters, this mistake may not become apparent until much later. This can be the cause of severe user frustration. To avoid this problem, the user interface program ought to verify the correctness of as many of the parameters as possible when they are entered. Of course, such foreground verification of parameters is not possible if the remote host to which the parameters apply is currently unreachable.

To explore the usefulness of background file transfer in the present Internet, we have implemented a file-mover service which we call the Background File Transfer Program or BFTP.

Section 2 describes BFTP and Section 3 presents our experience and conclusions. The appendices contain detailed information about the

user interface language for BFTP, a description of the program organization, and sample execution scripts.

2. Background File Transfer Program

2.1 General Model

In the present BFTP design, its user interface program and its FTC daemon program must execute on the same host, which we call the BFTP control host.

Through the user interface program, a BFTP user will supply all of the parameters needed to transfer a file from source host S to destination host D, where S and D may be different from the BFTP control host. These parameters include:

- o S and D host names,
- o login names and passwords on S and D hosts, and
- o S and D file names (and optionally, directories).

The user may also specify a number of optional control parameters:

* Source file disposition -- Copy, move (i.e., copy and delete), or simply delete the source file. The default is copy.
* Destination file operation -- Create/Replace, append to, or create a unique destination file. The default is create/replace ("STOR").
* FTP Parameters -- Explicitly set any of the FTP type, mode, or structure parameters at S and D hosts.
* Multiple Transfers -- Enable "wildcard" matching to perform multiple transfers.
* Start Time -- Set the time of day for the first attempt of the transfer. The default is "now" (i.e., make the first

attempt as soon as the request has been queued for the FTC daemon).

Finally, the user specifies a mailbox to which a completion notification message will be sent, and "submits" the request to the FTC daemon queue. The user can then exit the BFTP user

interface program.

If the transfer should fail permanently, the FTC daemon will send a notification message to the user's mailbox. In the event of a temporary failure (e.g., a broken TCP connection), the FTC daemon will log the failure and retry the transfer after some timeout period. The retry cycles will be repeated until the transfer succeeds or until some maximum number of tries specified has been reached. In either case, a notification message will then be sent to the user's mailbox.

The user can check on the progress of the transfer by reentering the BFTP user interface program, supplying a key that was defined with the request, and displaying the current status of the request. The user may then cancel the request or leave it in the queue.

The BFTP program includes a server-Telnet module, so it can be executed as a remotely-accessible service that can be reached via a Telnet connection to the BFTP well-known port (152). This allows a user on any Internet host to perform background file transfers without running BFTP locally, but instead opening a Telnet connection to port 152 on a BFTP service host. Of course, a user can also run the local BFTP user interface program directly on any host that supports it and for which the user has login privileges.

The next section discusses how BFTP uses standard FTP servers to perform the transfers, while the following section covers the user interface of BFTP.

2.2 File Transfer Mechanics for BFTP

The BFTP makes use of the "third party" or "Server-Server" model incorporated in the Internet File Transfer Protocol [RFC-959]. Thus, the FTC daemon opens FTP control connections to the existing FTP servers on source host S and destination host D and instructs them to transfer the desired file(s) from S to D. The S and D hosts may be any two Internet hosts supporting FTP servers (but at least one of them must support the FTP "PASV" command). This approach allows the implementation of a background file transfer

capability for the entire Internet at a very low cost.

Figure 1 illustrates the BFTP model of operation. Note that the BFTP control host is not necessarily the same as S or D. Figure 2 illustrates the FTP command interchange used in a typical Server-Server file transfer operation; this may be compared with the User-Server FTP scenario illustrated in Section 7 of RFC-959.

Since BFTP may be asked to transfer files between any two hosts in the Internet, it must support all the file types and transfer modes that are defined in RFC-959, not just a subset implemented by particular hosts.

BFTP supports the transfer of a set of files in a single request, using the standard technique:

(1) Send an NLST command to the source host S, specifying a pathname containing "wildcard" characters. The reply will contain a list of matching source file names.

(2) Execute a separate transfer operation for each file in this list. The destination file name in each case is assumed to be the same as the source file name; this requires that these names be compatible with the naming conventions of D.

It will typically be necessary to specify working directories for the transfers at S and D, so the file names will be simple, unstructured names on each system.

This approach depends upon the wildcard matching capability of the source host S. A more general implementation would acquire a complete list of the file names from the source host and do the matching in the FTC daemon, for example using a regular-expression matcher. Another useful extension would be a general pattern-matching file name transformation capability (e.g., like the one included in the 4.3BSD version of FTP) to generate appropriate destination pathnames for multiple requests.

Figure 1 -- BFTP Model of Operation

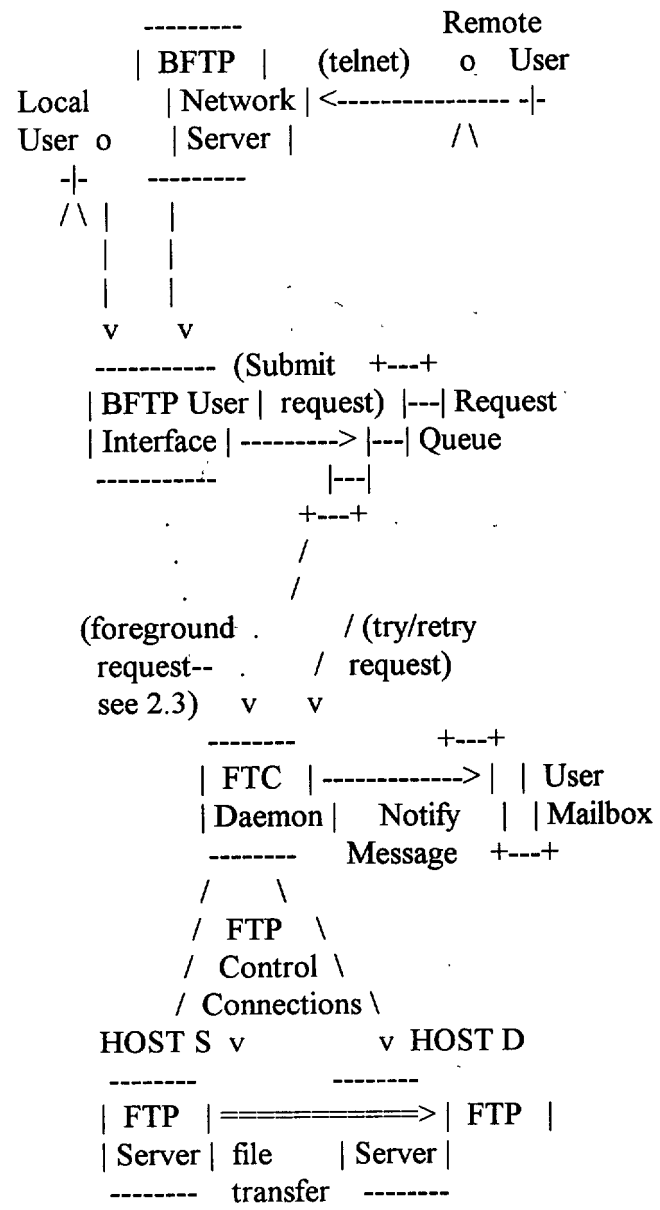
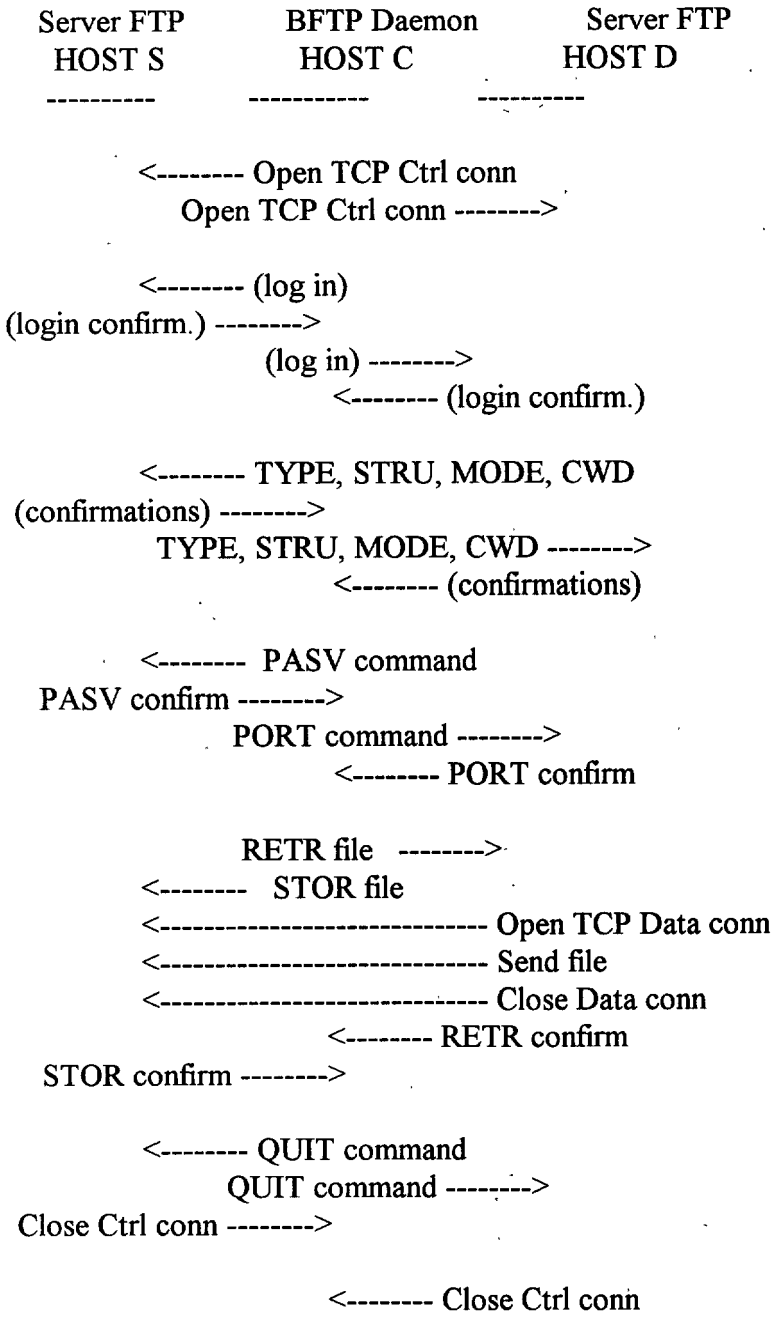


Figure 2 -- Server-Server File Transfer



BFTP currently utilizes the following Server-FTP commands [RFC-959]: USER, PASS, ACCT, PASV, PORT, RETR, STOR, STOU, CWD, NLST, MODE, STRU, TYPE, and QUIT.

The FTC daemon attempts to work around FTP servers that fail to support certain commands. For example, if a server does not support the optional command "CWD", the FTC daemon will attempt to construct a complete path name using the source directory name and the source file name. However, it is necessary that at least one of the two hosts support the FTP passive (PASV) command. While many FTP server implementations support do this command, some (in particular, the 4.2BSD FTP) do not. The PASV command was officially listed as being optional in RFC-959.

2.3 Reliable Delivery

The reliable delivery function of BFTP is analogous to reliable delivery in a transport protocol like TCP. Both depend upon repeated delivery attempts until success is achieved, and in both cases the choice of the retry interval requires some care to balance overhead against unresponsiveness.

Humans are impatient, but even their impatience has a limit. If the file cannot be transferred "soon", a human will turn to another project; typically, there is a tendency for the transfer to become less urgent the longer the wait. The FTC daemon of BFTP therefore starts each transfer request with a very short retry interval -- e.g., 10 minutes -- and then doubles this interval for successive retries, until a maximum interval -- e.g., 4 hours -- is reached. This is essentially the exponential backoff algorithm of the Ethernet, which is also used by transport protocols such as TCP, although BFTP and TCP have quite different rationales for the algorithm.

We must also define the meaning of reliable transmission for a multiple-transfer request. For example, the set of files selected by wildcard characters in a pathname is not well defined; the set may change while the request is pending, as files are created and deleted. Furthermore, it is unreasonable to regard the entire multiple transfer as a single atomic operation. Suppose that transferring a set of files fails part way through; for an atomic

operation, the files which had been successfully transferred would have to be deleted pending the next retry of the entire set. This would be ridiculously inefficient and may be impossible (since the communication path may be broken when it is time to issue the deletion requests).

BFTP addresses these issues in the following manner:

* For a multiple file operation, the FTC daemon saves the file name list returned by the first successful NLST command in the request queue entry. This name list determines the set of source files for the transfer; there can be no later additions to the set.

* The FTC daemon maintains a transfer status pointer. On each retry cycle, it tries to transfer only those files that have not already been successfully transferred.

* The request is complete when all the individual file transfers have been successful, a permanent failure has occured, or when the retry limit is reached.

* The notification message to the user lists the status of each of the multiple files.

2.4 BFTP User Interface

The purpose of BFTP is to simplify the file transfer process and to place the burden of reliability on the BFTP control host. We have attempted to provide a "user friendly" command interface to BFTP, similar in flavor to the user interface of the TOPS-20 operating system. This interface provides extensive prompting, defaulting, and help facilities for every command.

For a list of all BFTP commands, the user may enter "?<Return>" at the main BFTP prompt ("BFTP>"). Entering "help<Return>" and "explain<Return>" will provide increasing levels of explanatory material. To obtain information on a particular command, "help <command name><Return>" may be entered. The 'quit' or 'exit' command will exit from BFTP. Command and subcommand names may be abbreviated to the shortest unique sequence for that context; alternatively, a partial name can be automatically completed by typing <Return>.

The normal procedure for a BFTP user is to set up a set of parameters defining the desired transfer and then submit the

request to the FTC daemon. To give the user the maximum flexibility, BFTP supports three modes of submission:

o Background Operation

To request a reliable background file transfer, the user will issue the BFTP 'submit' command to the FTC daemon.

o Foreground Verification, Background Operation

The BFTP 'verify' command may be used to ascertain that file transfer parameters are valid. It causes BFTP to connect to the FTP servers on both the source and the destination hosts (if possible), log into both, verify the FTP parameters, and verify that the specified source file is present.

Once the 'verify' command has successfully completed, the user can issue the 'submit' command to schedule the actual file transfer.

o Foreground Operation

The BFTP 'transfer' command will perform the specified third-party transfer in foreground mode. This is illustrated by the dotted path bypassing the queue in Figure 1.

The easiest way to set up the parameters is to issue the 'prompt' command, which will prompt the user for all of the basic parameters required for most transfers. Certain unusual parameters must be set with the 'set' command (see Appendix B for details).

When entering any parameter, the following control characters may be used:

? will display help text for the parameter, indicating its meaning, the choices, and the default, and then reprompt for the parameter.

<ESC> will display the default value (or the last value set) for this parameter. The user can accept this default by entering <Return>, or else erase it with Control-W and enter a different value for the parameter, followed by <Return> to accept the entered value.

<Control-W>
will erase the value typed or displayed for current

parameter.

<Return>
will accept the value displayed for this parameter, and continue to the next parameter, if any. If the user has not typed a value or used <ESC> to display the default, <Return> will display the default and then accept it.

It is important to provide a means for a user to obtain status information about an earlier request or even to cancel an earlier request. However, these functions, especially cancellation, must be controlled by some user authentication. We did not want to build a user authentication database with each BFTP instance or require login to BFTP itself, and there is no Internet-wide user authentication mechanism. We adopted the following weak authentication mechanism as a compromise:

* When the 'submit' command is issued, it prompts the user for a character string called a "keyword", which recorded with the request.

* This keyword can be entered later as the argument to a 'find' command, which will display the status of all requests with matching keywords.

* Similarly, the keyword may be used to cancel the corresponding request.

If two different users happen to choose the same keywords, of course, this scheme will not protect each other's requests from accidental or malicious cancellation. However, a notification message will be sent at the time that a cancellation occurs.

To make a series of similar requests, the user needs only to change the individual parameters that differ from the preceding request and then issue a new 'submit' command, for each request. There are commands for individually setting each of the parameters that 'prompt' sets -- and 'time' -- to provide a shortcut for BFTP experts. A simpler but lengthier procedure is to use the 'prompt' command to run through the current set of parameters, reentering the parameters that must change and using the sequence <ESC><return> to retain the previous value for each of the others. The same procedures may be used to correct a mistake made in entering a particular parameter.

The current settings of all the BFTP parameters can be displayed at any time with the 'status' command, while the 'clear' command will return all parameters to their initial values. Finally, the 'request' command allows the user to save the current set of

parameters in a file or to restore the parameters from a previously-saved file.

There is also a window-based BFTP user interface for use on a Sun Workstation, described in Appendix A. The complete list of BFTP commands is presented in Appendix B.

3. Experience and Conclusions

BFTP has been available to users at ISI for some months. Users have reported a number of advantages of using BFTP:

(a) Some users prefer the prompting style of BFTP to the user interface of the foreground FTP they normally use.

(b) The BFTP "verify" command allows the user to verify that host names, passwords, and filenames are correct without having to wait for the entire transfer to take place.

(c) Since results are returned through the mail system, a transfer can occur without tying up a terminal line, a phone line, or even a window.

BFTP must be able to communicate with a variety of Server-FTP implementations, and we have observed much variation in the commands supported, error handling, and the timing in these servers. Some of the problems we have encountered are:

(1) Some systems (e.g., 4.2BSD) do not support the PASV command.

(2) 4.2/3BSD systems return a non-standard response to the NLST command. Instead of returning a list of complete path-names, they use an ad hoc format consisting of a directory name followed by a list of files.

(3) 4.2/3BSD systems may return a "permanent negative completion reply" (a 5xx FTP reply code) as a result of a communications failure such as a broken TCP connection. According to RFC-959, the appropriate response is a "transient negative completion reply" (a 4xx FTP reply code), which would inform the BFTP that the transfer should be retried.

(4) A number of servers return badly formatted responses. An example of this is the 4.2/3BSD response to an NLST command for a non-existent file name: an error string which is not preceded by a numerical response code.

To diagnose problems that do occur, we have found it very useful to have a complete record of the interchange between the FTC daemon and the two FTP servers. This record is saved and is currently always included in the notification message mailed to the user (see Appendix D for an example). As we get more experience with this program, some of the details of the transfer may be omitted from this log.

The use of library routines shared between modules makes it relatively easy to implement additional user interface programs. We are currently experimenting with a window version of BFTP, the "bftptool", which runs in the SunView environment, and is described in Appendix A. Some additional interfaces that might be useful are:

- o A command line interface for use in shell scripts and "Makefiles".
- o A more general library interface which would make it easy to invoke BFTP from a variety of programs.
- o Additional full-screen form based interfaces, for example a tool running in X-Window system environment.

Lastly, BFTP would benefit from the resolution of the following open protocol issues:

- o There currently exist no provisions for Internet-wide user authentication. In the BFTP context, this means that passwords required for a file transfer must be present in BFTP request files. The security of these passwords is subject to the limitations of the file system security on the BFTP control host. Anonymous file transfer provides a partial solution, but a more general, long term solution is needed.
- o Better mechanisms are needed to cope with the diversity of real file systems in the Internet.

  For example, an extension could be made to the FTP protocol to allow the daemon to learn the delimiter conventions of each host file system. This could allow a more flexible and powerful multiple-file facility in BFTP. This could include the automatic transfer of directory subtrees, for example.

4. References

[RFC-959] Postel, J., and J. Reynolds, "File Transfer Protocol (FTP)", RFC-959, USC/Information Sciences Institute,

Appendix A -- BFTP Implementation Structure

BFTP has been implemented on both a Sun workstation running Sun OS 3.4 (based on 4.2BSD) and a VAX running 4.3BSD. The program modules are: the local user interface programs "bftp", the Internet server program "bftpd", and the FTC daemon "fts". BFTP makes use of the "at" command, a UNIX batch job facility, to submit requests and execute the daemon. An additional user interface program, the "bftptool", is available for Sun OS 3.4, and runs in the SunView environment.

BFTP keeps its state in a set of control files: request files, command files, and message files. These files are stored in the home directory specified for the environment of the process running "bftp". If a user is running "bftp" directly, this will typically be the user's home directory. In the case where a user has made a Telnet connection to the well-known port 152 on a BFTP service host, "bftp" is started by "bftpd" (or "inetd", indirectly). As a result, the control files will be owned by the user-id under which "inetd" was started, normally "root", and stored in the top level directory "/". Note, however, that under BFTP all user files are written by the FTP servers, which are presumed to enforce the operating systems' access control conventions. Hence, BFTP does not constitute a system integrity exposure.

A.1 User Interface Program

The BFTP user interface program "bftp" may be run directly via a UNIX shell. Once the program has been started, the prompt "BFTP>" will appear and commands may be entered. These commands are described in detail in Appendix B.

A.2 Tool-Style User Interface Program

The BFTP user interface program "bftptool" may be started from a shell window in the SunView environment on a Sun workstation. The BFTP commands may be selected via the left mouse button. The various file transfer parameters appear in a form-style interface; defaults and multiple-choice style parameter values can be filled in via menus. An advantage of this form-style interface program is that it is possible to view all of the file transfer parameters

simultaneously, providing the user with a sense for which parameter values might be mutually exclusive.

Help information can be displayed in a text subwindow by positioning the on-screen mouse pointer over a command or a parameter, and clicking the center mouse button. (No standard mechanism for displaying help information is currently included in

the SunView package.)

The commands used in the "bftptool" are for the most part very similar to the commands described in Appendix B. Request submittal and the execution of the FTC daemon are identical for the "bftp" and the "bftptool" interface programs.

A.3 Internet Server

The Internet server program "bftpd" can be invoked by opening a Telnet connection to a well-known port, and does not require login. The "bftpd" program runs under "inetd", the standard BSD4.x well-known port dispatcher. When a SYN arrives for the BFTP well-known port, "bftpd" opens the TCP connection and performs Telnet negotiations. It then passes control to the user interface "bftp" which allows the user to enter file transfer requests.

A.4 BFTP Server Daemon

The BFTP file transfer control daemon program is named "fts" (for "File Transfer Service"). This module contains code to actually cause a single file transfer operation using the FTP server-server model as shown in Figures 1 and 2. It is invoked with the command "fts <request-file>". The <request-file> contains the necessary parameters for the file transfer, in ASCII format, separated by linefeeds. Such a request file may be created by the user interface program, "bftp".

As a byproduct of the development of BFTP, "fts" represents a server-server FTP driver that can be run independent of the "bftp" program. Parameters used in the file transfer are read from a request file, which is created and accessed via library routines which can be shared between modules. This could be used to perform FTP's under program control.

Appendix B: BFTP Command Summary

B.1 Special Editing Characters

In the "bftp" program, the special editing characters for command words, subcommands, and parameter fields are as follows:

<return>	Accept current command/field.
<escape>	Complete current command/field, or display default.
<space>	Complete and delimit current command.
<delete>	Erase last character.
control-L	Refresh screen.
control-R	Refresh line.
control-U	Erase line.
control-W	Erase current token.
?	List legal options.

B.2 BFTP Commands

The remainder of Appendix B consists of a list of the BFTP commands. Each command should be followed by a carriage-return. In the description of the syntax for each command, square brackets "[]" are used to indicate a ssubcommand, or a list of possible subcommands, which are separated by the "|" character. Angle brackets "<>" are used to indicate a description of a parameter where the choices would be too numerous to list, for example "<host name/number>".

B.2.1 Clear Command

Return all parameters to their default values.

clear

B.2.2 Destination Commands

Set the destination directory.

ddir <directory name>

Set the destination file name.

dfile <file name>

Set the destination host, user, and password.

dhost <host name/number> <login> <password>

B.2.3 Explain Command

Display a short explanation of how to use BFTP.

explain

B.2.4 Find Command

Find and display a previous request.

find

BFTP will prompt for the request id, which is printed when the request is first submitted. An example of a request id is "bftp583101774". BFTP also prompts for the request keyword, which was determined by the user when the request was first submitted. If no keyword was specified, a <CR> should be typed. If no request id is entered, BFTP will display all requests which contain a matching keyword.

RequestID (optional): <bftp-request-id>
RequestKeyword: <keyword>

After BFTP has displayed a summary of a matching request, it asks whether the request is to be changed, or canceled.

Do you wish to change this request? [yes | no]
Do you wish to cancel this request? [yes | no]

If the user indicates that the request is to be changed, BFTP will read in the parameters and cancel the existing request. At this point the user may make any desired changes and use the "submit" command to requeue the request. At this point a new request id will be assigned and displayed.

Although this may happen extremely rarely, if at all, it is possible that a system crash (or the interruption of the BFTP program) at a particularly inopportune moment may leave a request which is not queued. When the "find" command locates such a request, it displays the warning:

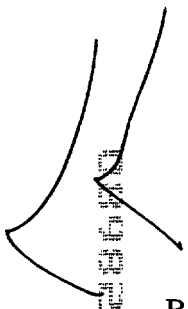
```
Your request is NOT currently queued.
```

If this happens, the request may be read in and resubmitted using the following procedure:

```
Your request is NOT currently queued.
Do you wish to change this request? yes

 (BFTP displays the parameters that have been read in.)

Previous request canceled.
Use the 'submit' command to submit a new request.
```



B.2.5 Help Command

Print local help information.

```
help
help <command>
```

B.2.6 Quit Command

Clear parameters and exit the BFTP program.

```
quit
```

B.2.7 Prompt Command

Prompt for commonly-used parameters.

```
prompt
```

The following are the parameters that BFTP prompts for:

```
    copy/move/delete: [copy | move | delete]
    ascii/ebcdic/image/local:
        [ascii|ebcdic] [nonprint|telnet|carriage-control]
or
        [image]
or
        [local] <byte size>
(see "set type" for additional information)
```

```
    Source --
      Host: <host name/number>
      User: <login>
      Password: <password>
      Dir: <directory including a delimiter, e.g., "/" or ">">
          (either an absolute path, or relative to the login)
      File: <file name>

    Destination --
      Host: <host name/number>
      User: <login>
      Password: <password>
      Dir: <directory>
      File: <file name>
```

Once the prompting has been completed, the current values of all parameters will be displayed. Parameters not mentioned in the prompting will be initialized with default values, and may be changed via the "set" commands.

B.2.8 Request Commands

The request commands enable the user to save a set of BFTP parameters in a "request-file" for future use. Subcommands are provided to to list all available request-files, or to read, write, or delete a request-file. All request-files are stored in the user's home directory. Therefore, this facility is not available when the user is accessing BFTP by telneting to port 152.

Delete request file "bftp-save.name".

request delete <name>

List all bftp-save files.

request list

Read a request file in as the current request.

request load <name>

Save the current request in a file named "bftp-save.name".

request store <name>

B.2.9 Set Commands

The "set" commands have complex subcommand structures and are used to set many of the less commonly used FTP parameters. The subcommands of "set" are as follows:

Set the account for the source/destination login.

set account [source | destination] <account string>

Set to true to append to destination file.

set append [true | false]

The source file will be copied to the destination file name.

set copy

The source file will be deleted after the file has been moved or copied.

set delete

Set the mailbox to which the results will be returned. The mailbox should be in standard internet format, for example: "deschon@isi.edu".

set mailbox <mailbox string>

Set the FTP transfer mode.

set mode [stream | block | compress]

The source file will be deleted after it has been copied.

set move

Set to true to transfer multiple files.

set multiple [true | false]

Set the port for the source/destination FTP connection.

set port [source | destination] <port number>

Set the FTP structure.

set structure [file | record | page]

Set the FTP type and format / byte size parameters. Note that a normal text file is usually "ascii", and a "binary" file is often the same as an "image" file.

set type [ascii|ebcdic] [nonprint|telnet|carriage-control]
or
set type [image]
or
set type [local] <byte size>

Set to true if the STOU command is to be used. If the STOU command is supported by the destination host, the file will be stored into a file having a unique file name.

set unique [true | false]

Set to true to display full FTP conversations for "verify" and "transfer" commands.

set verbose [true | false]

B.2.10 Source Commands

Set the source directory.

sdir <directory name>

Set the source file name.

sfile <file name>

Set the source host, user, and password.

shost <host name/number> <login> <password>

B.2.11 Status Command

Display the current parameter values.

status

B.2.12 Submit Command

Submit the current request for background FTP.

submit

BFTP prompts for the following information:

StartTime: <date and/or time>
ReturnMailbox: <internet mailbox>
RequestKeyword: <made-up keyword>

B.2.13 Time Command

Set the start time, the starting retry interval, and the maximum number of tries.

time <date and/or time> <minutes between tries>
<maximum number of tries>

B.2.14 Transfer Command

Perform the current request in the foreground.

transfer

B.2.15 Verify Command

Make the connections now to check parameters.

```
verify
```

Appendix C Example BFTP User Script

```
deschon.isi edu 1% telnet hobgoblin.isi.edu 152
Trying 128.9.0 42 ...
Connected to hobgoblin.isi.edu.
Escape character is '^]'

BFTP Server (hobgoblin.isi.edu)

Background File Transfer: For help, type '?', 'help', or 'explain'.

BFTP> prompt

Copy/Move/Delete: copy

Source --
   Host: deschon.isi.edu
   User: deschon
   Password:
   Dir: ./
   File: foo*

Destination --
   Host: venera.isi.edu
   User: deschon
   Password:
   Dir: ./temp/
   File: foo*

StartTime: Tue Oct  6 10:14:43 1987 (interval) 60 (tries) 5
ReturnMailbox. deschon@isi.edu
RequestPassword:

BFTP> set multiple true
BFTP> status
   Request type: COPY

   Source --
      Host 'deschon.isi.edu'
      User 'deschon'
      Pass SET
```

set multiple files

need to define the host again

```
    Acct  ''
    Dir  '/'
    File  'foo*'
    Port. 21

  Destination --
    Host  'venera.isi.edu'
```

```
User 'deschon'
Pass: SET
Acct: ''
Dir: '/temp/'
File 'foo*'
Port: 21

Structure: file, Mode: stream, Type: ascii, Format: nonprint
Multiple matching: TRUE
Return mailbox: 'deschon@isi.edu', Password: SET
Remaining tries: 5, Retry interval: 60 minutes

Start after Tue Oct  6 10:14:43 1987.

BFTP> submit
Checking parameters.

Request bftp560538880 submitted to run at 10:14 Oct 6.

BFTP> quit
bye
Connection closed by foreign host.
deschon.isi.edu 2%
```

Appendix D Sample BFTP Notification Message

```
Received-Date: Tue, 6 Oct 87 10:15:52 PDT
Date: Tue, 6 Oct 87 10:15:47 PDT
From: root (Operator)
Posted-Date Tue, 6 Oct 87 10:15:47 PDT
To: deschon
Subject: BFTP Results: bftp560538880

Request bftp560538880 submitted to run at 10:14 Oct 6.

 Tue Oct 6 10 15:22 1987: starting...

  Request type: COPY
  Source deschon.isi.edu-deschon-XXX--21-./-foo*
  Destination: venera.isi.edu-deschon-XXX--21-./temp/-
  Stru: F, Mode: S, Type: A N, Creation: STOR
  Multiple matching: TRUE
  Return mailbox: 'deschon@isi.edu', Password: SET
  Remaining tries: 5, Retry interval: 60 minutes

Connect to deschon.isi.edu, 21
deschon.isi edu ==> 220 deschon.isi.edu FTP server (Version 4.7
                    Sun Sep 14 12:44:57 PDT 1986) ready.
Connect to venera.isi.edu, 21
venera.isi.edu ==> 220 venera.isi.edu FTP server (Version 4.107
                   Thu Mar 19 20:54:37 PST 1987) ready.
deschon.isi.edu <== USER deschon
deschon.isi.edu ==> 331 Password required for deschon.
deschon.isi edu <== PASS XXX
deschon.isi.edu ==> 230 User deschon logged in.
venera.isi.edu <== USER deschon
venera.isi.edu ==> 331 Password required for deschon.
venera.isi.edu <== PASS XXX
venera.isi edu ==> 230 User deschon logged in.
deschon.isi edu <== CWD ./
deschon.isi edu ==> 200 CWD command okay.
venera.isi edu <== CWD ./temp/
venera.isi.edu ==> 250 CWD command successful.
deschon.isi edu <== PORT 128,9,1,56,4,106
deschon.isi edu ==> 200 PORT command okay.
```

```
deschon.isi.edu <== NLST foo*
deschon.isi.edu ==> 150 Opening data connection for /bin/ls
                    (128.9.1.56,1130) (0 bytes).
deschon.isi.edu ==> 226 Transfer complete.
deschon.isi.edu <== PASV
deschon.isi.edu ==> 502 PASV command not implemented.
venera.isi.edu <== PASV
```

```
venera.isi.edu ==> 227 Entering Passive Mode (128,9,0,32,6,200)
deschon.isi edu <== PORT 128,9,0,32,6,200
deschon.isi edu ==> 200 PORT command okay.
deschon.isi edu <== RETR foo
venera.isi.edu <== STOR foo
deschon.isi edu ==> 150 Opening data connection for foo
                    (128.9.0.32,1736) (0 bytes).
deschon.isi edu ==> 226 Transfer complete.
venera.isi edu ==> 150 Openning data connection for foo
                    (128.9.1.56,20).
venera.isi edu ==> 226 Transfer complete.
venera.isi edu <== PASV
venera.isi edu ==> 227 Entering Passive Mode (128,9,0,32,6,201)
deschon.isi edu <== PORT 128,9,0,32,6,201
deschon.isi edu ==> 200 PORT command okay.
deschon.isi edu <== RETR foo1
venera.isi.edu <== STOR foo1
deschon.isi edu ==> 150 Opening data connection for foo1
                    (128.9.0.32,1737) (4 bytes).
deschon.isi edu ==> 226 Transfer complete.
venera.isi edu ==> 150 Openning data connection for foo1
                    (128.9.1.56,20).
venera.isi edu ==> 226 Transfer complete.
deschon.isi edu <== QUIT
venera.isi edu <== QUIT

Tue.Oct 6 10 15:39 1987: completed successfully.
```

connect again to the host for the second file